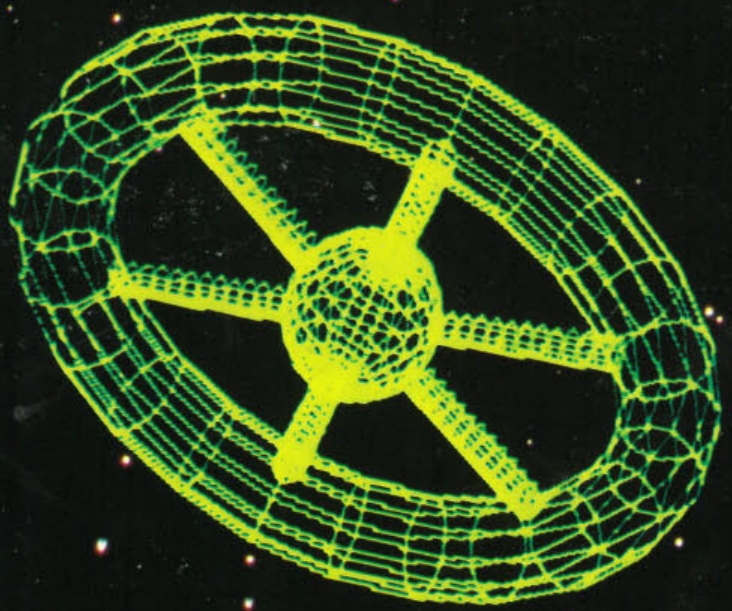
INPUT

Publicación práctica para usuarios de

commodore

Revista mensual 1987 Precio 375 Ptas Año 2 Número 16



COMO DIBUJAR EN 3-D

MAS SOBRE INTELIGENCIA ARTIFICIAL

GRAFICOS ANIMADOS

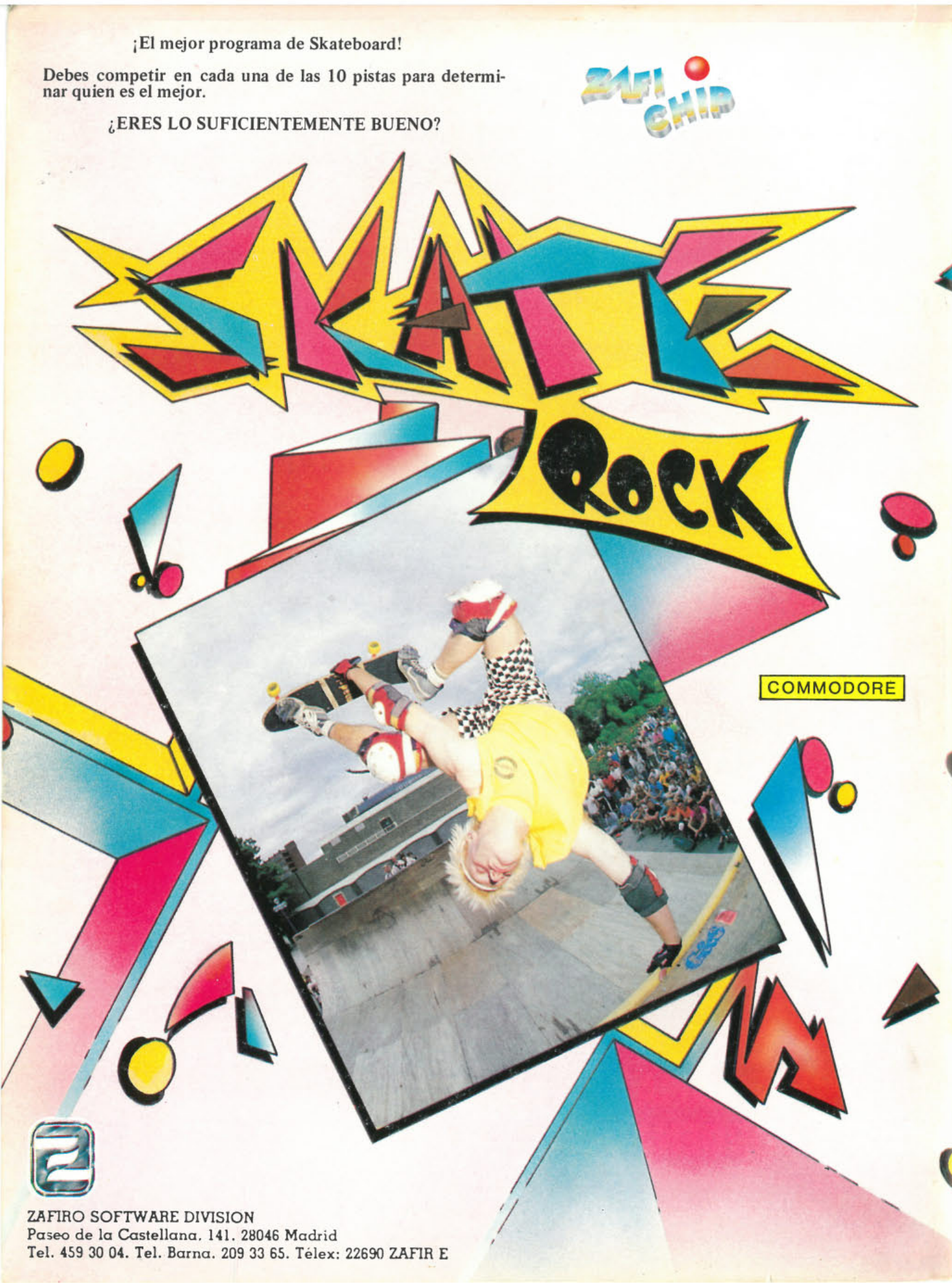
APRENDE MECANOGRAFIA CON TU ORDENADOR

Participa en el concurso de aplicaciones

¡El mejor programa de Skateboard!

Debes competir en cada una de las 10 pistas para determinar quien es el mejor.

¿ERES LO SUFICIENTEMENTE BUENO?



ZAFIRO SOFTWARE DIVISION
Paseo de la Castellana, 141. 28046 Madrid
Tel. 459 30 04. Tel. Barna. 209 33 65. Télex: 22690 ZAFIR E

AÑO 2 NUMERO 16

DIRECTOR: Manuel Pérez
COORDINADOR EDITORIAL: Francisco de Molina
DIRECTOR DE ARTE: Luis F. Balaguer
REALIZACION GRAFICA: Diego Tudela
COLABORADORES: José M.ª Gils, Christophe Païs, Antonio Díaz, Jaime Mardones, Equipo Molisoft
FOTOGRAFIA: Joan Boada

INPUT Commodore es una publicación de PLANETA-DE AGOSTINI, S.A.
GERENTE DIVISION DE REVISTAS: Sebastián Martínez

PUBLICIDAD: Grupo Jota
Madrid: c./ General Varela, 35
Teléf. 270 47 02/03
Barcelona: Avda. de Sarriá, 11-13, 1.º
Teléf. 250 23 99

FOTOMECANICA: TECFA, S.A.

IMPRESION: Sirven Grafic
C/ Gran Vía, 754-756, 08013 Barcelona
Depósito legal: B. 38.114-1986

SUSCRIPCIONES: EDISA,
López de Hoyos, 141. 28002 Madrid
Teléf. (91) 415 97 12

REDACCION:
Aribau, 185, 1.º
08021 Barcelona

DISTRIBUIDORA:
R.B.A. PROMOTORA DE EDICIONES, S.A.
Travesera de Gracia, 56. Edificio Odiseus.
08006 Barcelona.

El precio será el mismo para Canarias que para la Península y en él irá incluida la sobretasa aérea.

INPUT Commodore es una publicación controlada por

INPUT Commodore es independiente y no está vinculada a Commodore Business Machines o sus distribuidores.

INPUT Commodore no mantiene correspondencia con sus lectores, si bien la recibe, no responsabilizándose de su pérdida o extravío.

© 1987 by Planeta-De Agostini, S.A.

Copyright ilustraciones del fondo gráfico de Marshall Cavendish

INPUT commodore

SUMARIO

EDITORIAL 4

PROGRAMACION
APRENDE A DIBUJAR EN 3.D (I) 6
IDEAS PARA TUS TITULARES 15
REFINA TUS GRAFICOS 51

INTELIGENCIA ARTIFICIAL
LISP (I) 22

CODIGO MAQUINA
MOVIMIENTO EN EL VIC-20 46

APLICACIONES
MECANOGRAFIA POR ORDENADOR 39

PARTICIPA
PROYECTO GIOTTO (II) 55

REVISTA DE SOFTWARE 60

EL ZOCO DE INPUT 66

PROGRAMACION DE JUEGOS (COLECCIONABLE) 31
ADIVINA MIS PALABRAS
ULTIMANDO EL JUEGO DE LAS PALABRAS

SACALE PARTIDO A TU ORDENADOR

Después de la compra del nuevo ordenador o de haber completado tu equipo con nuevos periféricos, ha llegado el momento de que te pongas a trabajar en serio con ellos y deshechar la tentación fácil de aparcarlos en un rincón. Como siempre, INPUT te ofrece la posibilidad de combinar el aprendizaje de las técnicas de programación y la información de las novedades de hardware, con el juego, la enseñanza de trucos útiles y el seguimiento de los más interesantes programas que se publican.

Conscientes de que el mundo de la informática vive un movimiento permanente, del que los medios de comunicación no debemos prescindir, hemos confeccionado una encuesta, con cuyas respuestas esperamos poder sintonizar mejor con vosotros. A fin de que os resulte más sugestiva, se efectuará un sorteo entre todas las contestaciones recibidas.

Por último, hemos de señalar que con la publicación de este número de INPUT inauguramos un nuevo *Concurso de Aplicaciones y Utilidades*, destinado a intentar ampliar la gama de servicios que os prestan vuestros ordenadores.

A nuestro juicio, esta nueva competición estimulará un importante intercambio de experiencias y permitirá introducir nuevos temas en vuestra revista.

Sin duda, vuestras respuestas serán tan entusiastas como en anteriores ocasiones, lo que elevará la calidad y el interés de la revista.

Así pues a DISFRUTAR de vuestro INPUT y... a PARTICIPAR.

LOS MEJORES DE INPUT

Hemos pensado que es interesante disponer de un ***ranking*** que ponga en claro, mes a mes, cuáles son los programas preferidos de nuestros lectores. Para ello, es obligado preguntaros directamente y tener así el mejor termómetro para conocer vuestras preferencias. Podéis votar por cualquier programa aunque no haya sido comentado todavía en **INPUT**.

El resultado de las votaciones será publicado en cada número de **INPUT**.

Entre los votantes sortearemos 10 cintas de los títulos que pidáis en vuestros cupones.

Nota: No es preciso que cortéis la revista, una copia hecha a máquina o una simple fotocopia sirven.

Enviad vuestros votos a: **LOS MEJORES DE INPUT** Aribau, 185. Planta 1. 08021 Barcelona

ELIGE TUS PROGRAMAS

Primer título elegido _____ **Segundo título elegido** _____

Tercer título elegido _____ **Programa que te gustaría conseguir** _____

Qué ordenador tienes _____ **Nombre** _____

1.er Apellido _____ **2.º Apellido** _____

Fecha de nacimiento _____ **Teléfono** _____

Dirección _____ **Localidad** _____

Provincia _____

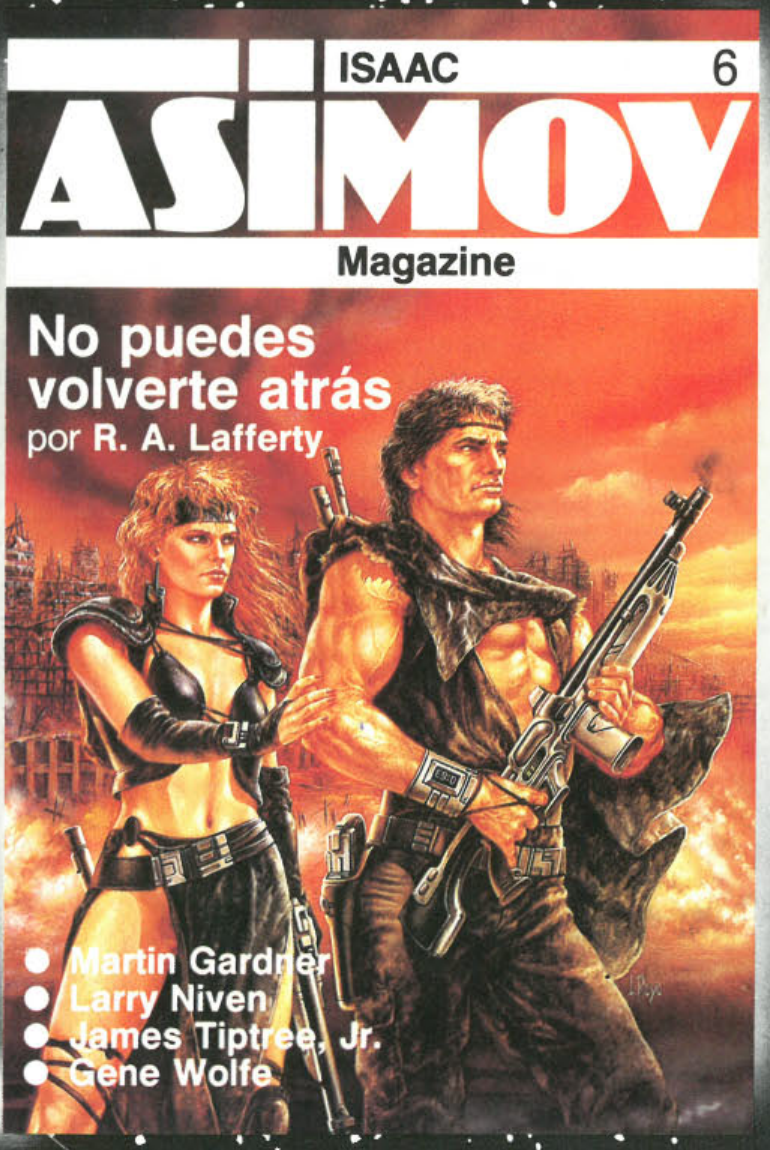
INPUT COMMODORE N.º 16

NUEVA
REVISTA MENSUAL

isaac ASIMOV

selecciona para ti los mejores relatos de

CIENCIA FICCION



A-062

APRENDE A DIBUJAR EN 3-D (I)

En este número os ofrecemos el primero de una serie de artículos sobre dibujos de estructuras de alambre mediante rejillas y círculos. En posteriores números de INPUT podrás ver cómo se convierten en bloques construidos en tres dimensiones.

El diseño y concepción de dibujos de «estructuras de alambre», utilizando los gráficos generados en un ordenador, se ha impuesto en muchos campos, especialmente en el publicitario, gracias a que permite trabajar a la imaginación sin las limitaciones que conlleva el método clásico del papel y el lápiz. Si tú, como usuario de un ordenador doméstico, has deseado generar imágenes animadas de este tipo en la pantalla de tu televisor debes haberte quedado decepcionado al creer que tu micro no puede igualar el esplendor de las imágenes publicitarias producidas conjuntamente por programadores, fotógrafos y artistas. Sin embargo no todo está perdido, porque en esta serie de artículos puedes aprender a dibujar y manipular imágenes en tres dimensiones de una forma que muchos ingenieros de diseño podrían envidiar.

La capacidad de tu ordenador para poder direccionar puntos individuales de pixels (elementos de imagen) en la pantalla proporciona un medio potente para poder realizar dibujos, como habrás podido apreciar en varios artículos ya publicados en INPUT.

El método normal para dibujar líneas en sistemas gráficos controlados por ordenadores es similar a cómo utilizas un lápiz y un papel. Puedes mover el cursor sobre la pantalla sin marcarlo, o puedes desplazarlo por la pantalla dejando una señal. Cambiar de colores es tan simple como cambiar de lápiz. La principal diferencia entre el sistema de un ordenador y una persona que utiliza un lápiz es que el ordenador es más rápido y es mejor dibujando líneas rectas, una tarea que se hace difícil para la mayoría de la gente, si no emplea una regla.

Esta capacidad de dibujar líneas te permite dibujar también imágenes de perfiles o contornos e ilustraciones del tipo «estructura de alambre» de objetos tridimensionales. Los gráficos de estructuras tramadas o de alambre, son una representación imaginaria que consiste en una rejilla de líneas dispuesta sobre la superficie del objeto. Por lo general no se intenta ocultar las líneas detrás del objeto.

Para poder hacer esto se requiere una considerable cantidad de materiales y de trabajo. El resultado es el de que el objeto aparece como si estuviera hecho de una estructura de alambre unida para formar una rejilla.

Este tipo de visualización puede ser animada girando y moviendo el objeto de estructura de alambre, como se puede observar, por ejemplo, en las imágenes publicitarias de coches y juguetes. La animación rápida no puede hacerse sólo con un ordenador doméstico, ya que cada fotograma tiene que ser visualizado a un ritmo de, por lo menos, 25 fotogramas por segundo, para una animación continua, y un ordenador doméstico tarda mucho más de 1/25 de segundo para las visualizaciones más simples. En realidad, la mayor parte de los visualizadores comerciales animados por ordenador no están generados en «tiempo real» (como aparecen en la pantalla). En vez de ser así, cada fotograma se genera por separado, empleando segundos, minutos e incluso horas, para visualizaciones complicadas de alta resolución, y después son grabadas en magnetoscopios o filmadas para poder ser visualizadas, después, a la velocidad correcta.

Para las visualizaciones estáticas el tiempo no es lo importante, aunque puedes volverte un poco impaciente esperando a que la imagen sea completada. Sin embargo, con frecuencia es igualmente interesante, si no más, poder observar una imagen de estructura de alambre mientras es construida en la pantalla, al igual que lo es poder ver el objeto terminado, especialmente en los dibujos más complejos.

Para esta serie de artículos, los usuarios del Commodore 64 necesitaréis el cartucho BASIC de Simons para poder ejecutar estos programas.

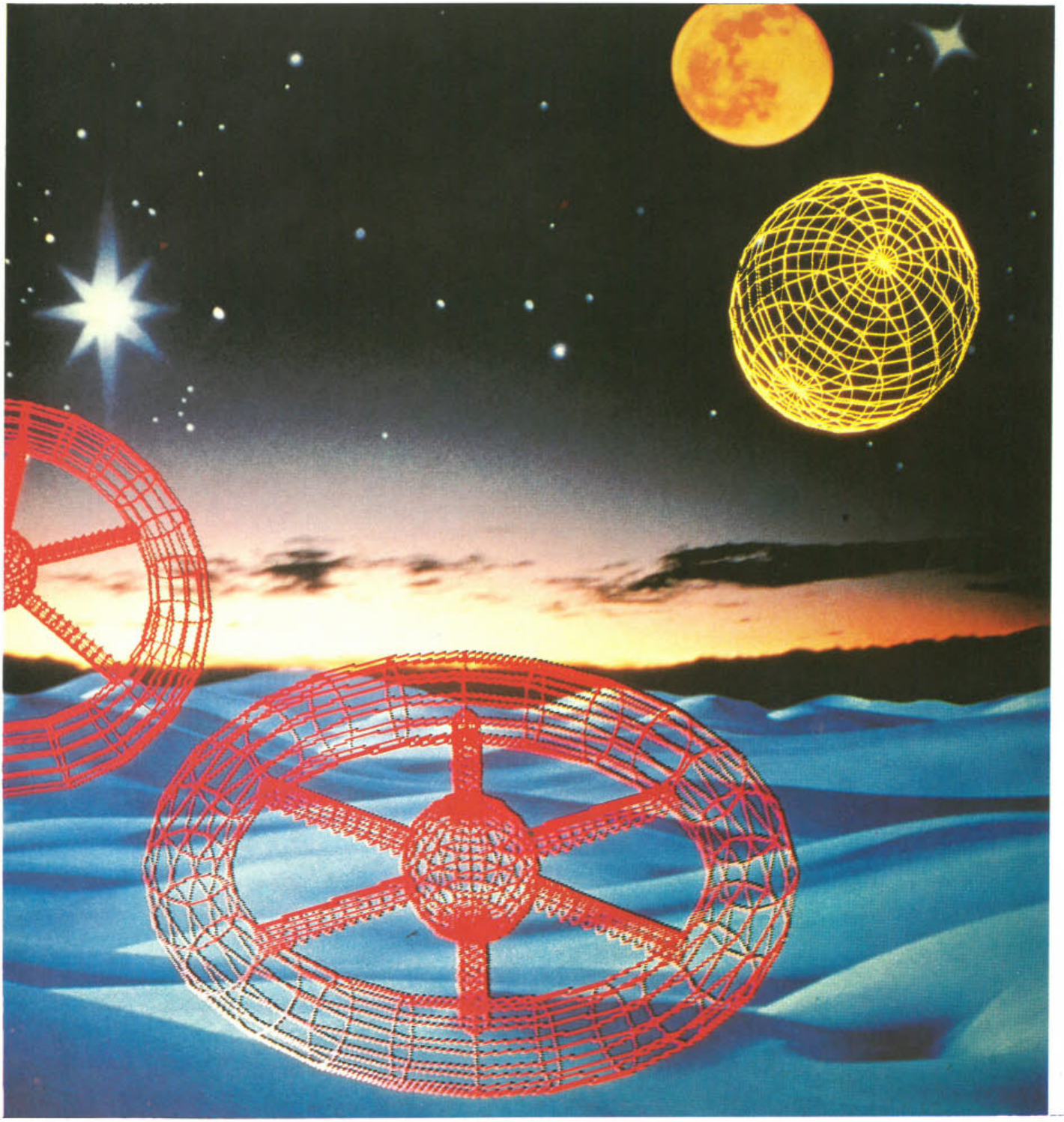
Es mejor empezar a dibujar formas simples —como un cubo o una esfera— y una vez que se haya ganado experiencia, se puede intentar dibujar formas más complejas e interesantes. En este artículo aprenderás cómo se pueden generar algunas formas bidimensionales y hacer que aparezcan en un espacio tridimensional.

VALORES INICIALES

Incluso para un ordenador de alta velocidad y de buena capacidad para dibujar líneas, se requiere una gran cantidad de trabajo para producir una imagen, bien elaborada, de «estructura de alambre». Se necesita, esencialmente, una orden para mover el cursor, sin que éste dibuje nada, y otra orden para dibujar desde la última posición del cursor a una nueva.

Al principio de un programa de gráficos se dispone el ordenador para ser utilizado bajo el modo de gráficos y se limpia la pantalla. Es interesante estructurar el programa de tal forma que todas las órdenes para dibujar sean agrupadas en subrutinas. Además de ser una práctica de programación de sonido, este acercamiento metódico sirve para prevenir el tener que rees-

- DIBUJOS DE ESTRUCTURAS DE ALAMBRES
- DIBUJANDO EN TRES DIMENSIONES
- PROYECCION ISOMETRICA
- UNION DE REJILLAS PARA CONSTRUIR UN CUBO
- UTILIZANDO EL PROGRAMA PARA DIBUJAR OTRAS FORMAS



cribir secciones del programa que obtienen los mismos resultados. La estructuración dilata el programa, pero esto no es importante para las presentaciones estáticas y la gran flexibilidad que ofrecen.

Para componer la primea etapa de estas rutinas de dibujos teclea esta sección de programación, pero no hagas RUN todavía, porque es incompleta.

```
9ØØØ PRINT"[SHIFT+
     CLR/HOME]"
9Ø3Ø RETURN
```

Ahora, cuando quieras dibujar, no necesitarás decirle al ordenador cómo mover el cursor o cómo marcar la pantalla para configurar cada forma diferente. En lugar de esto es mucho más simple definir cada forma nueva en términos de estas rutinas básicas.

P y R

¿Puedo añadir color para destacar mis dibujos de estructura de alambre?

Por lo general los dibujos de estructuras de alambre se hacen sólo en dos colores, normalmente en blanco y negro. Una de las razones para esto es que muchos colores complicarían la imagen.

Sin embargo, la cuestión más importante es que la adición de colores reduce la resolución de la imagen y en algunos ordenadores puede causar problemas cuando un color se sobreimpresiona sobre otro. En el Commodore 64, el empleo de color habría dividido la pantalla de dibujo, y el programa hubiera tenido que ser modificado para compensarlo. Hacia el final de esta serie de artículos hablaremos sobre el empleo de los colores, pero no hay ninguna razón para que no puedas comenzar a experimentar con colores desde este mismo momento.

DIBUJO DE LINEAS

La forma más simple para dibujar es la línea, así que aquí tienes una rutina para hacerla (todavía no podrás hacer RUN con el programa).

```
95ØØ LINEXS,YS,XE,YE,1
955Ø RETURN
```

Esta rutina especifica una posición de inicio, coordenadas XS, YS, y una posición de final, con las coordenadas XE, YE para la línea.

La rutina de dibujo de líneas es la base de la mayor parte de los programas de dibujo o diseño de estructuras de alambre. Con estas líneas se establece uno de los «bloques de construcción» básicos en el proceso de diseño: la rejilla.

DIBUJANDO UNA REJILLA

Para representar una superficie y cualquier irregularidad que ésta pueda contener, como hendiduras, colinas o valles, es mejor visualizarla no como un área continua, encerrada en un rectángulo, sino como una rejilla de líneas horizontales y verticales. Cualquier característica irregular en la superficie puede ser mostrada como distorsión de estas líneas. Ahora pasaremos a la siguiente sección del programa, pero sin hacer RUN, para definir una rutina que permita dibujar una rejilla.

```
5ØØØ JA=LW/NX
5Ø1Ø XS=XA
5Ø2Ø FOR JB=Ø TO NX
5Ø25 YS=YA:XE=XS:YE=YA+LH
5Ø3Ø GOSUB 95ØØ
5Ø4Ø XS=XS+JA
5Ø5Ø NEXT JB
5Ø6Ø JA=LH/NY
5Ø7Ø YS=YA+LH
5Ø8Ø FOR JB=Ø TO NY
5Ø9Ø XS=XA+LW:XE=XA:YE=YS
51ØØ GOSUB 95ØØ
511Ø YS=YS-JA
512Ø NEXT JB
513Ø RETURN
```

Las coordenadas (XA,YB) especifican la esquina izquierda del fondo de la rejilla. LW especifica el ancho y LH la altura. NX especifica el número de divisiones horizontales, y NY el número de divisiones verticales. La variable JA especifica la distancia entre las líneas verticales y el bucle FOR...NEXT dibuja líneas horizontales escalonadas en esta distancia. El segundo bucle FOR...NEXT dibuja líneas verticales en pasos calculados por la línea 5060.

La rutina entre las líneas 5000 y 5180, dibuja líneas horizontales en un paso desde la izquierda a la derecha, y líneas verticales en un paso desde el fondo hasta arriba, ya que una rejilla de este tipo sólo puede trazar superficies planas. Por ejemplo, no puede representar irregularidades junto con la superficie.

Para poder ver la rejilla en la pantalla teclea estas líneas, que llaman a la rutina, y ejecuta RUN.

```
1ØØ HIRES Ø,1:COLOUR 5,1
175 XA=Ø:YA=Ø:LW=32Ø:LH=
    2ØØ:NX=4:NY=4
18Ø GOSUB 5ØØØ
19Ø GOTO 19Ø
```

Ahora ves una rejilla que ocupa toda la pantalla. Para ver cuán versátil es el programa, realiza estos cambios:

```
175 XA=1Ø:YA=1Ø:LW=3ØØ:LH
    =18Ø:NX=1:NY=1
```

En esta ocasión se dibuja una caja rectangular porque se ha especificado una rejilla con una división horizontal y una vertical. Si se proporcionan valores apropiados a NX y NY, como arriba, se puede dibujar una rejilla en la que el número de divisiones horizontales sea diferente del número de divisiones verticales.

Realiza ahora nuevos cambios:

```
175 XA=1Ø:YA=8Ø:LW=15Ø:LH
    =9Ø:NX=15:NY=1Ø
```

La rejilla ya no llena la pantalla y en su lugar encontramos un cuadrado en la parte inferior izquierda del área. La

forma del cuadrado se obtiene proporcionando los valores apropiados a LW y LH, y el número de divisiones está especificado por NX y NY, igual que arriba.

DIBUJAR CIRCULOS

Las rejillas rectangulares no son los únicos «bloques de construcción» que pueden ser utilizados para dibujar estructuras de alambre; con frecuencia, es bastante útil poder dibujar círculos. Algunos ordenadores poseen controles que te permiten dibujar círculos si se especifica el centro y el radio.

Sin embargo, estas órdenes directas no permiten el grado de control que se necesita para poder dibujar imágenes tridimensionales. Con la perspectiva, un círculo observado desde un punto podría parecer una elipse desde otro, o semejar una curva. Por lo tanto, es mejor poder definir una función general.

Una de las maneras de dibujar un círculo es la de emplear una serie de secciones pequeñas de líneas rectas. Partiendo de la base de que las líneas son cortas, la circunferencia del círculo aparecerá como una curva continua, pero cuanto más pequeñas sean las líneas, se necesitarán más y por lo tanto se prolongará el tiempo de dibujar. Aquí presentamos una rutina para dibujar un círculo de radio R, con el centro en (XS,YS); pero no ejecutes el programa todavía:

```
6ØØØ IF N=Ø
     THENN=N+INT(R/1Ø)
6Ø2Ø JA=2*π/N
6Ø5Ø XR=XS:YR=YS
6Ø6Ø JB=Ø:XS=XS+R
6Ø7Ø FOR JC=2 TO N
6Ø8Ø JB=JB+JA
6Ø9Ø XE=XR+R*COS(JB):YE=YR
     +R*SIN(JB):GOSUB
     95ØØ
61ØØ XS=XE:YS=YE
611Ø NEXT JC
612Ø XE=XR+R:YE=YR:GOSUB
     95ØØ
613Ø XS=XR:YS=YR
616Ø RETURN
```

La variable N dispone el número de segmentos de línea que deben ser utilizados para la circunferencia del círculo. Si especificas N=0, la línea 6000 calcula cuántos segmentos se necesitan para el círculo más uniforme, teniendo en cuenta el tamaño de la visualización en pantalla.

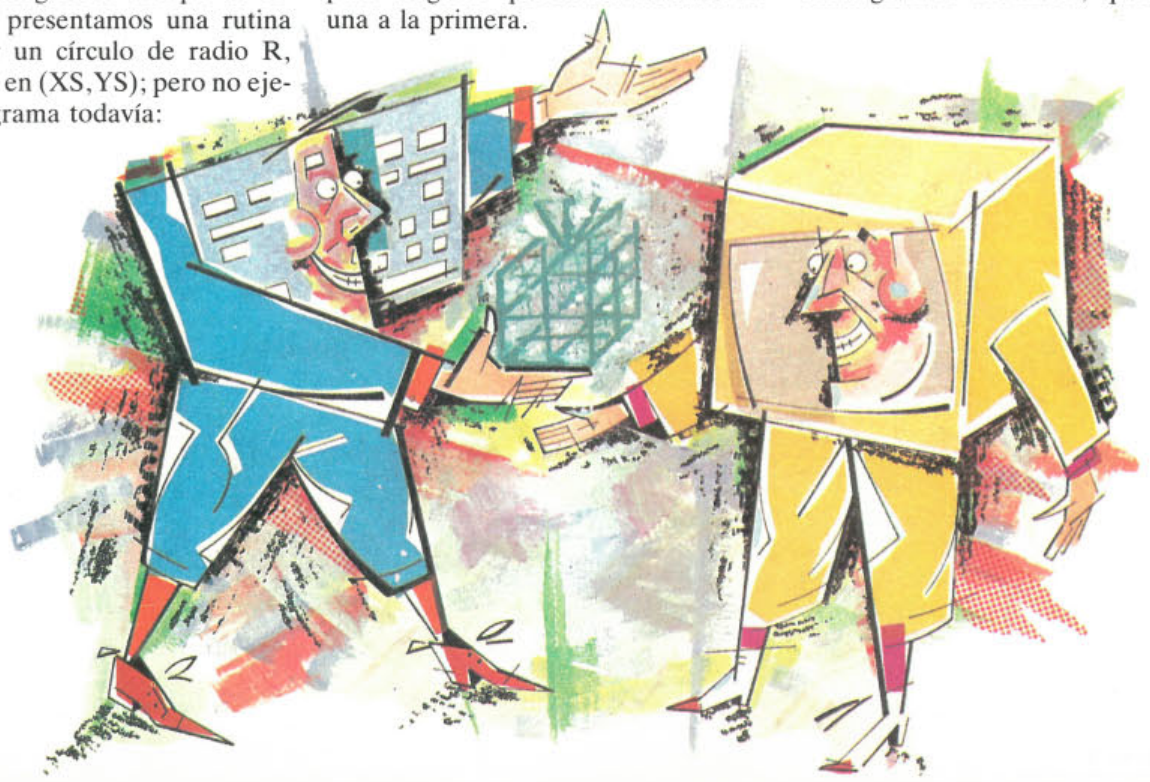
La línea 6020 calcula el ángulo de cada segmento de línea en la circunferencia. La línea 6050 mueve el cursor a una posición de la circunferencia. El bucle FOR...NEXT dibuja cada segmento de línea, excepto el último, que es dibujado por la línea 6120 para asegurar que la última línea se una a la primera.

Para observar cómo funciona la rutina, borra la línea 180 y añade las siguientes líneas para llamarla:

```
15Ø FOR R=2Ø TO 1ØØ
    STEP 2Ø
155 XS=16Ø:YS=1ØØ:N=24
16Ø GOSUB 6ØØØ
17Ø NEXT R
```

La presentación en tu pantalla debe mostrar ahora círculos concéntricos en el centro de la misma. Al igual que con la rutina de rejilla, puedes variar los parámetros del programa que llaman a la rutina para cambiar la presentación. La línea 150 asigna el radio del primer círculo y la cantidad en que es incrementado para darle el radio de círculos sucesivos. La línea 155 especifica el centro de los círculos y el número de las secciones de línea en la circunferencia. A modo de prueba puedes modificar estos valores y observar el efecto en la presentación.

Si estás preguntándote qué es lo que ha pasado con la rutina de la rejilla, hay que decir que permanece en la memoria, pero, desde que has escrito las líneas de código que llaman a la rutina, el ordenador no puede mostrarlas. Rutinas como éstas pueden ser reunidas en una biblioteca de útiles rutinas gráficas de líneas, que pueden



ser empleadas cuando sea necesario. Pueden ser utilizadas en cualquier tipo de programas, así como en dibujos de estructuras de alambre. También se pueden añadir nuevas rutinas en la medida en que se requiera. Una vez que estén en el ordenador hay que proceder a almacenarlas en cinta o disco. Cuando desees utilizarlas, vuelve a cargarlas en la memoria, y si es necesario, toma varias rutinas.

En la próxima oportunidad podrás ver cómo se utilizan estas rutinas para crear dibujos de estructuras de alambre en tres dimensiones.

CONVENCIONES VISUALES

En teoría, por lo menos, no tiene sentido decir que uno puede crear un objeto tridimensional (por ejemplo una caja) en un medio de dos dimensiones, ya sea en un trozo de papel o en la pantalla del televisor. Pero lo que *sí* se puede hacer es emplear una convención visual de tal modo que nuestros ojos puedan percibir o «leer» una serie de líneas como representación de un objeto sólido.

La pintura de un paisaje, por ejemplo, aparenta tener una profundidad y una distancia porque el artista ha utilizado un truco visual: la perspectiva. La perspectiva depende del hecho de que nosotros comprendamos que los objetos se hagan pequeños cuando se mueven en la distancia, y que las líneas paralelas parecen converger cuando se encuentran más alejadas del observador.

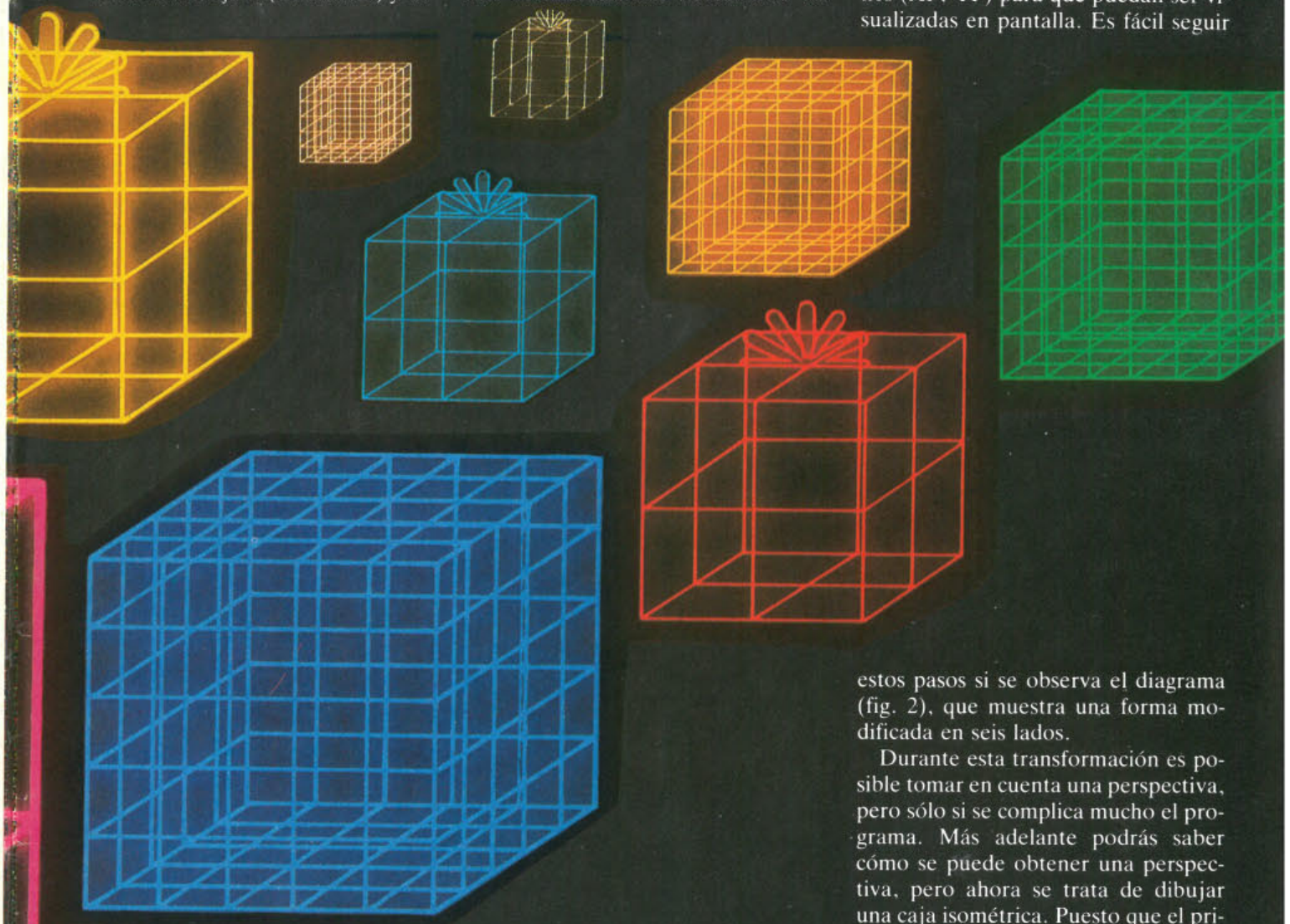
La perspectiva no es la única convención o regla que se utiliza en este tipo de representación. En los dibujos técnicos es habitual emplear una convención denominada proyección isométrica. En este caso, las líneas que se alejan desde la posición del observador son dibujadas en un ángulo, como en un dibujo en perspectiva.

Sin embargo, a diferencia de los dibujos en perspectiva estas líneas no convergen, ni los objetos se hacen pequeños a medida que se alejan. Esto tiene muchas ventajas en las aplicaciones técnicas, como pueden ser el tomar medidas directas de cualquier línea ya que su escala no está afectada por esta dirección.

Aunque se puede representar una

perspectiva en una visualización por ordenador, como veremos más adelante, es más difícil de crear que una proyección isométrica. Los dibujos isométricos se basan simplemente en componer un tercer eje en la pantalla.

Ya tienes un eje X (horizontal) y un eje Y (vertical). Todo lo que necesitas es un tercer eje, el Z, que está en un ángulo asignado a los dos primeros. Cualquier línea que sea dibujada en este ángulo se entiende que representa a una línea que va desde (o hacia) el observador. Mira la figura 1.



TRANSFORMANDO EL PROGRAMA

Utilizando esta convención o regla, una de las formas de producir imágenes de estructuras de alambre en tres dimensiones es a partir de dos formas de dos dimensiones, empleando rutinas como las generadas para las rejillas y los círculos, tal y como hemos visto más arriba. Algunas de estas formas bidimensionales necesitarán ser distorsionadas para ajustarse al ángulo del eje Z. Esto requiere una serie de transformaciones de las coordenadas basadas en complicados principios matemáticos. Pero no te preocupes, tu ordenador puede hacerlo fácilmente.

Lo primero que debes hacer es transformar las coordenadas de dos dimensiones (X, Y) a coordenadas tridimensionales (X', Y', Z') en el plano deseado. Esto significa que tienes que especificar a qué distancia del eje Z debería estar el plano. Después necesitas transformar las coordenadas en 3-D (X', Y', Z') en un nuevo grupo de coordenadas 3-D (XE, YE, ZE) basadas en la posición y dirección desde la que estás observando al objeto. Finalmente, tienes que transformar estas coordenadas 3-D (XE, YE, ZE) en coordenadas de dos dimensiones (XP, YP) para que puedan ser visualizadas en pantalla. Es fácil seguir estos pasos si se observa el diagrama (fig. 2), que muestra una forma modificada en seis lados.

Durante esta transformación es posible tomar en cuenta una perspectiva, pero sólo si se complica mucho el programa. Más adelante podrás saber cómo se puede obtener una perspectiva, pero ahora se trata de dibujar una caja isométrica. Puesto que el primer y el último grupo de coordenadas son bidimensionales, y todas las transformaciones son lineales (las líneas rectas permanecen rectas, y las líneas paralelas permanecen paralelas), y ya que no hay que realizar ajustes para la perspectiva, puedes lograr estas transformaciones cambiando las rutinas originales de Inicialización y de Dibujo.

Si has almacenado una copia del programa que explicábamos en la primera parte del artículo, cárgalo y después teclea las líneas siguientes. Es una buena idea almacenar el listado

completo porque necesitarás llamar algunas de las rutinas en la medida en que el programa se desarrolle en artículos sucesivos. Si no almacenaste el programa, teclea las líneas 5000 a 5130. Estas líneas forman la rutina para dibujar una rejilla.

Ahora teclea las siguientes líneas, pero no hagas RUN todavía:

```
9ØØØ PRINT"[SHIFT
     +CLR/HOME]"
9Ø3Ø XX=1
9Ø4Ø YY=1
9Ø7Ø RETURN
95ØØ Q1=XS*XX+XY*YS+XO+
     16Ø:Q2=YX*XS+YY*YS+
     YO+1ØØ: Q3=XE*XX+XY*
     YE+XO+16Ø
95Ø1 Q4=YX*XE+YY*YE
     +YO+1ØØ:LINEQ1,Q2,Q3,
     Q4,1
955Ø RETURN
```

La rutina de Inicio desde la línea 9000 hasta la línea 9070, pone en los valores iniciales la transformación de variables (XX, XY, YX, YY, XO, YO) a valores que hacen que la rutina de dibujo trabaje como antes.

La rutina mueve los ejes de tal forma que el origen esté en el centro de la pantalla. Se traslada el punto de origen porque es conveniente pensar en los ojos del observador como situados directamente sobre el origen en el centro de la pantalla. Este punto se verá con más claridad en el próximo artículo, cuando puedas ver cómo se puede cambiar el punto de observación y cómo se puede añadir perspectiva.

Para poder ver el efecto de cada una de estas variables de transformación, y para tener una mejor percepción de lo que sucede, cambia las líneas 100 a 180, como se señala abajo, para llamar a la rutina de Rejilla. Borra la línea 155 y teclea:

```
1ØØ COLOUR 5,1
11Ø GOSUB 9ØØØ
13Ø XA=-4Ø:YA=-2Ø:LW=4Ø:
    LH=4Ø:NX=5:NY=5
135 GOSUB 5ØØØ
14Ø XA=Ø:YA=-2Ø:LW=4Ø:LH=
    4Ø:NX=1Ø:NY=1Ø
145 GOSUB 5ØØØ
15Ø GET A$:IF A$="" THEN 15Ø
16Ø NRM:INPUT"ENTRAR XX,
    XY,YX,YY,XO,YO";XX,XY,YX,
    YY,XO,YO
17Ø HIRESØ,1:GOTO 13Ø
```

Ejecuta el programa para poder ver dos pequeñas rejillas cuadradas en el centro de la pantalla, con una línea que te solicita que introduzcas XX, XY, YX y así sucesivamente. Los usuarios de Commodore deben pulsar una tecla. Éstas son las variables de transformación.

XX e YY colocan los factores de escala en las direcciones horizontal y vertical. Si se les da valores negativos se produce una reflexión cerca del eje. XO e YO realizan el balance para situar las rejillas en diferentes posiciones en la pantalla. XY e YX permiten las rotaciones y distorsiones (divididas) de la imagen. Las variables de transformación tienen los siguientes valores iniciales: XX=1, XY=Ø, YX=Ø, YY=1, XO=Ø y YO=Ø. Para transformar la imagen necesitas dar a estas variables valores diferentes, y ésta es la razón de la expresión INPUT en la línea 160. A continuación te presentamos un grupo de valores para que puedas hacer una prueba: tecléalos separados por una coma, después pulsa ENTER o RETURN.

-1,Ø,Ø,1,Ø,Ø provoca una reflexión cerca del centro del eje vertical de la pantalla.

Ø,5,Ø,Ø,2,Ø,Ø gradúa la imagen en dos, verticalmente, y a la mitad, horizontalmente.

1,Ø,Ø,1,1ØØ,-5Ø mueve la imagen 100 unidades a la derecha y 50 unidades hacia abajo o arriba.

Ø,-1,1,Ø,Ø,Ø hace rotar la imagen 90 grados en dirección contraria a las agujas del reloj cerca del centro de la pantalla.

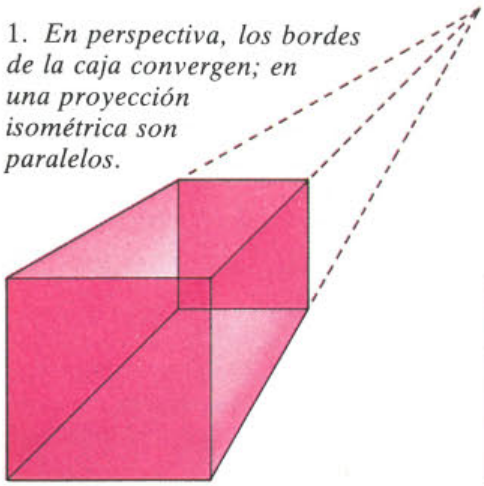
1,Ø,5,Ø,1,Ø,Ø divide la imagen horizontalmente.

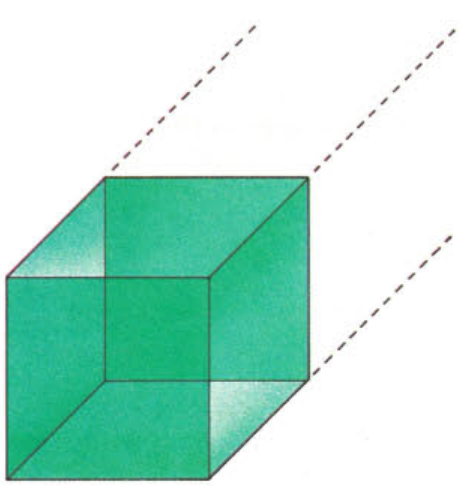
1,Ø,5,Ø,5,1,Ø,Ø divide las imágenes tanto horizontal como verticalmente.

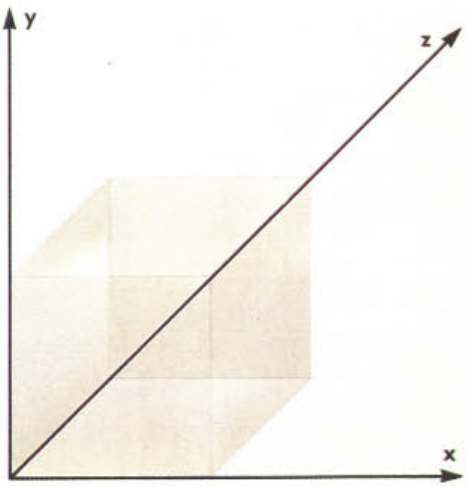
Para poder predecir los resultados con precisión necesitas entender matrices de aritmética. Así que aunque puedas variar la forma de esta imagen, deberás saber cómo se pueden combinar los valores y las rejillas para obtener una imagen 3-D completa. Y esto es precisamente lo que hace el siguiente programa de acuerdo a los procedimientos que se indican a continuación.

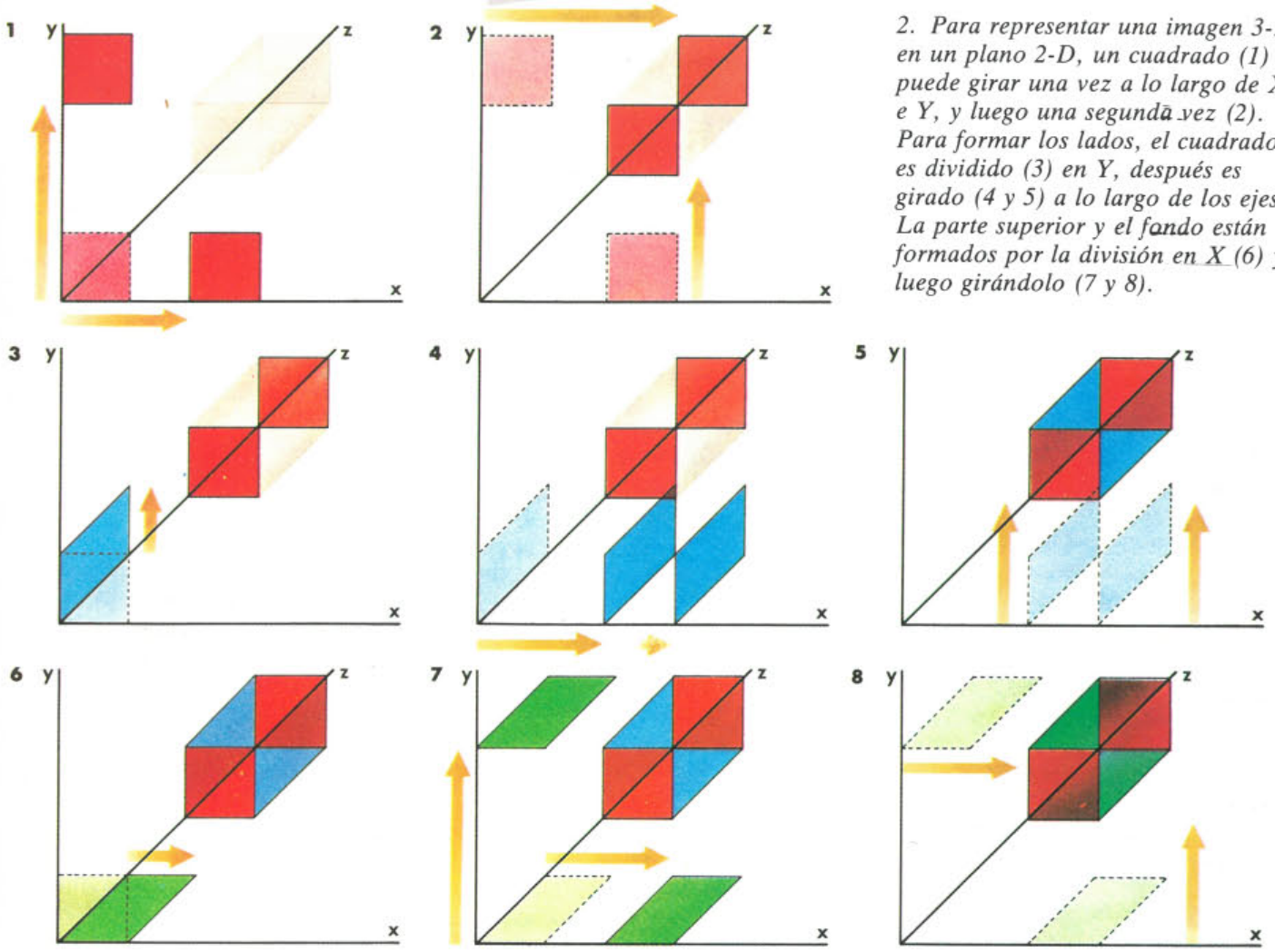
Borra las líneas 135, 145 y 170 y teclea:

1. *En perspectiva, los bordes de la caja convergen; en una proyección isométrica son paralelos.*









2. Para representar una imagen 3-D en un plano 2-D, un cuadrado (1) puede girar una vez a lo largo de X e Y, y luego una segunda vez (2). Para formar los lados, el cuadrado es dividido (3) en Y, después es girado (4 y 5) a lo largo de los ejes. La parte superior y el fondo están formados por la división en X (6) y luego girándolo (7 y 8).

```
1ØØ HIRES Ø : COLOUR 5,1
```

seguido de las líneas 110 a 520, siguientes:

```
11Ø GOSUB 9ØØØ
12Ø L=12Ø:N=5
13Ø DX=.45*L:DY=.3*L
14Ø XN=-(L+DX)/2
15Ø YN=-(L+DY)/2
16Ø XO=XN
17Ø YO=YN
18Ø GOSUB 5ØØ
19Ø XO=XN+DX
2ØØ YO=YN+DY
21Ø GOSUB 5ØØ
22Ø XY=DX/L:XO=XN
23Ø YY=DY/L:YO=YN
24Ø GOSUB 5ØØ
25Ø YO=YN+L
26Ø GOSUB 5ØØ
27Ø XX=DX/L:XY=Ø
28Ø YX=DY/L:YY=1:YO=YN
29Ø GOSUB 5ØØ
3ØØ XO=XN+L
31Ø GOSUB 5ØØ
32Ø GOTO 32Ø
5ØØ XA=Ø:YA=Ø:LW=
    L:LH=L:NX=N:NY=N
51Ø GOSUB 5ØØØ
52Ø RETURN
```

Cuando se pulsa RUN y se ejecuta el programa se debe ver un cubo en el centro de la pantalla. Todos los lados están dibujados por la misma rutina, que es, simplemente, la que se utiliza para dibujar una rejilla con variables apropiadas para transformarla en la forma correspondiente. Pero se han de colocar otras dos variables antes de que el cubo pueda ser dibujado. Éstas son la variable L, que marca el tamaño del cubo, y la variable N, que dispone el número de cuadrados de rejilla. La variable DX especifica hasta qué punto de la izquierda o de la derecha de la parte posterior del cubo es desplazada desde el frente, y DY especifica cuál es la extensión de desplazamiento de arriba hacia abajo de los mismos lados. DX y DY aseguran, además, que el cubo aparezca proporcionado.

Otro par de variables —XN en la línea 140 e YN en la línea 150— sitúan balances para la posición de inicio de la parte frontal del cubo. Estas compensaciones son trasladadas a XO en la línea 160 y a YO en la línea 170, luego se puede dibujar la parte posterior del cubo (en la línea 180).

La parte frontal y la parte posterior del cubo son idénticas, así que para dibujar el frente del cubo el único requisito es incrementar la compensación en la dirección X (línea 190) y en

la dirección Y (línea 200). Estas compensaciones alinean la posición de la parte posterior con los lados que has dibujado.

P y R

¿Puedo cambiar este programa para dibujar imágenes diferentes?

La rutina básica en este programa es la que dibuja la rejilla. Utiliza la sección que dibuja líneas, que podrías utilizar en un nuevo programa tuyo para, por ejemplo, dibujar triángulos. Esto no diferiría demasiado de la rutina de rejilla, pero necesitarías más atención si hicieras el intento de llamarla varias veces para dibujar una forma como la de tetraedro —una figura de cuatro lados triangulares—. Para dibujar la base, los valores dados para la transformación de variables serían los mismos de la caja, pero tendrías que abandonar el dibujo del lado que no necesitas, es decir, el de la parte superior de la caja. Una tarea más complicada es cambiar las variables de transformación (líneas 270 a 300) para dar un ángulo a la rejilla triangular y situarla para poder dibujar los tres lados.

Es mucho más simple cambiar el programa para hacer que dibuje cajas de diferentes tamaños y formas no cerradas. Por ejemplo, cambia el valor de L en la línea 120 para variar el tamaño de la caja, y cambia al mismo tiempo el valor de N, en la misma línea.

A grandes valores de N, la caja es virtualmente sólida. Para abandonar el dibujo de los lados frontal y posterior, borra la línea 311 (que dibuja un arco en la caja) e inserta: 155 GOTO 22Ø, después RUN.

La línea 240 dibuja la parte superior, pero antes que nada hay que transformar la rejilla —XY e YY— y las compensaciones deben ser redefinidas en las líneas 220 y 230. La misma vista de la rejilla se emplea para la parte del fondo, que sólo necesita ser graduada y posicionada (línea 250) antes de ser dibujada. De igual forma, desde las líneas 270 a 310, transforman la rejilla para dibujar el lado derecho, y luego vuelven a su posición para dibujar el lado izquierdo.

Todas las rejillas cuadradas de cada lado son de la misma forma y tamaño y todas las líneas paralelas en cada rejilla permanecen paralelas.

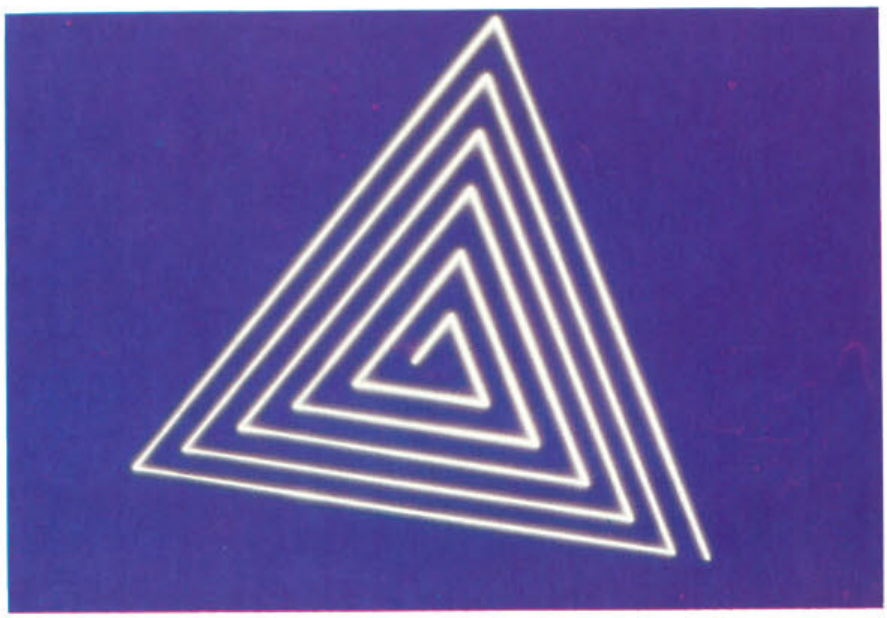
La razón se encuentra en que no se ha hecho todavía ningún ajuste de perspectiva. Más aún, es difícil adaptar el programa para mostrar otras vistas del cubo.

Más adelante veremos cómo hacer dibujos de estructuras de alambre en perspectiva, que pueden ser observados desde cualquier posición. Mientras tanto, cambia las líneas 120 y 130, abajo, y añade estas pocas líneas para envolver el cubo como si fuera un regalo. No te olvides de pulsar SAVE y hacer una copia:

```
12Ø  L=6Ø:N=2
13Ø  DX=Ø.6*L:DY=Ø.5*L
311  GOSUB 1ØØØØ
1ØØØØ  SS=16Ø:TT=7Ø
1ØØ1Ø  FOR M=Ø TO π STEP
       Ø.Ø2
1ØØ2Ø  D=COS(6*M)
1ØØ3Ø  S=D*COS(M):T=D*SIN(M)
1ØØ4Ø  S=S*24:T=ABS(T*2Ø)
1ØØ5Ø  LINE SS,TT,16Ø+S,
       7Ø-T,1
1ØØ55  SS=16Ø+S:TT=7Ø-T
1ØØ6Ø  NEXT M
1Ø13Ø  RETURN
```

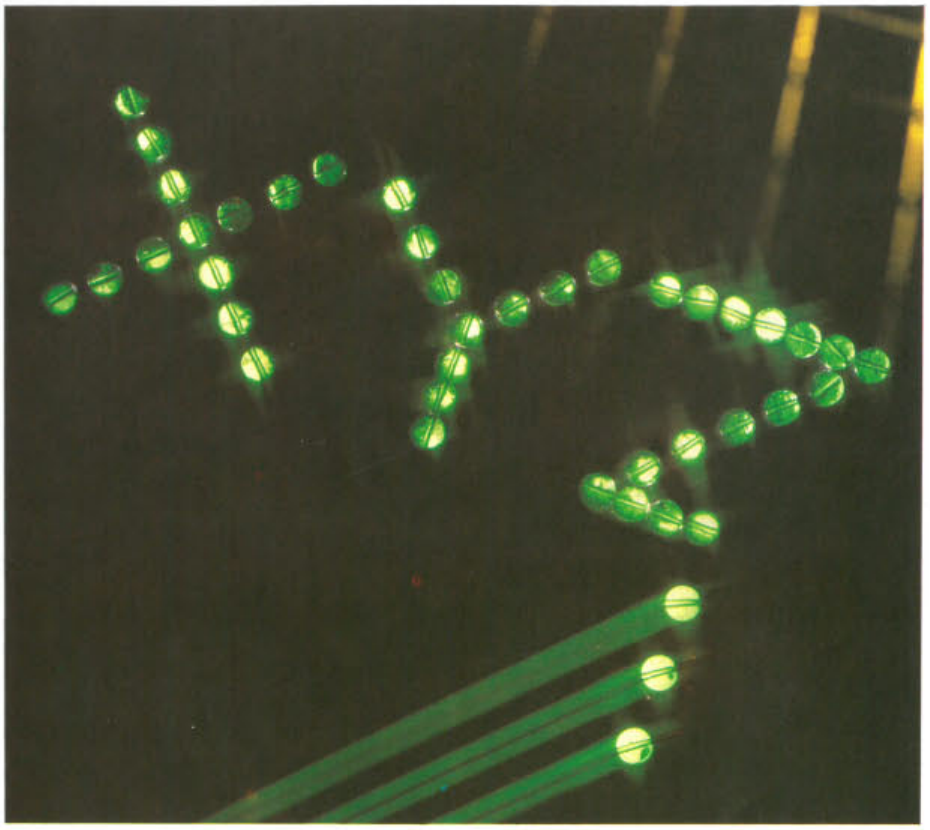
DIBUJOS FUERA DE PANTALLA

Si cambias la línea 120, de tal forma que L tenga valores más grandes, seguramente te encontrarás que el programa se bloquea y te da un mensaje de error. Esto sucede porque algunos ordenadores no pueden dibujar una línea si uno de los puntos especificados está fuera del alcance de la pantalla. En teoría, si la imagen está por entero en la pantalla, no deben presentarse problemas. El programa no está limitado por este problema, ya que en un próximo artículo se hace una revisión para asegurar que la máquina dibuje sólo las líneas que son válidas. Gracias a ello, el programa no es interrumpido con mensajes de error.



MAS IDEAS PARA TUS TITULARES

- DIBUJO DE LETRAS LINEA A LINEA
- CAMBIO DE LA ESCALA DE LOS CARACTERES
- INTRODUCCION DE LAS PALABRAS
- DISEÑA TU PROPIA TIPOGRAFIA



En la primera parte de este artículo vimos la forma de crear letras en pantalla utilizando el conjunto de caracteres gráficos. Aquí te proporcionamos nuevo material tipográfico más unas cuantas ideas sobre el empleo de todas las nuevas teorías.

A continuación analizaremos una nueva forma de crear letras para hacer presentaciones en pantalla, utilizando los gráficos de alta resolución para dibujarlas línea a línea con ayuda de la sentencia DRAW.

Los programas que siguen funcionan de una manera muy parecida a los de los bloques gráficos de letras, en el sentido de que cada uno de ellos consiste en una serie de sentencias DATA que indican al ordenador de qué manera tiene que construir cada letra. Esta información se almacena en un *array*, y tú puedes introducir las palabras que quieras ensanchar en forma de cadena de caracteres. Igual que en aquel caso, las letras se van construyendo con arreglo a las instrucciones que el ordenador va encontrando, y se van representando en pantalla. Posteriormente verás de qué manera puedes utilizar esta presentación (o la que proceda de cualquier otro generador de letras) como parte de tus propios programas.

TIPOGRAFIA A LA MEDIDA

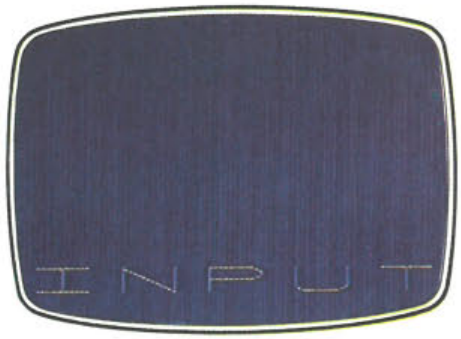
El uso de líneas con alta resolución te permite una libertad mucho mayor en el estilo de tus letras que los métodos anteriores. La ampliación del conjunto de caracteres sólo está limitada por el doble de la altura y de la anchura de los caracteres estándar. Las letras de los bloques gráficos están completamente fijadas por su diseño en las sentencias DATA.

Las sentencias DATA de los siguientes programas también fijan de qué manera se construye cada letra (así, por ejemplo, para una L, el ordenador recibe instrucciones para trazar una línea vertical y otra horizontal). Pero todas estas instrucciones están relacionadas unas con otras, por lo que no determinan el aspecto global de la letra. Piensa como si cada letra estuviera encerrada en una caja imaginaria. Si dicha caja es alta y estrecha, obtendrás una letra alta y estrecha, mientras que si es baja y ancha, la letra resultante será baja y ancha. Los programas están escritos de tal forma que puedes decidir estos factores de escala al principio.

Teclea ya las líneas del programa.

```
 5 PRINT "[SHIFT+CLR/HOME]
   UN MOMENTO..."
1ØØ FORT=8192TO16192:
   POKET, Ø: NEXT
11Ø DIML(26,2Ø)
115 T=1: X=Ø
14Ø READA: X=X+1: L(T,X)=A
145 IFA=1Ø THEN X=Ø: T=T+1
15Ø IFT>26 THEN 16Ø
155 GO TO 14Ø
157 PRINT "ESPERA POR
   FAVOR...":
   FORT=8192TO12ØØØ:
   POKET, Ø: NEXT
16Ø X=1Ø: Y=3Ø: PRINT
   "ENTRA FACTOR
   X(ANCHURA)"
17Ø INPUTX1: IFX1>4Ø THEN
   16Ø
18Ø PRINT "ENTRA FACTOR Y
   (ALTURA)"
19Ø INPUT Y1: IF Y1>4Ø THEN
   18Ø
2ØØ PRINT "INTRODUCE LA
```

```
    PALABRA"
21Ø INPUT A$
213 FORT=1Ø24TO2Ø23:
    POKET, 1: NEXT
215 POKE 53265, PEEK
    (53265) OR 32: POKE
    53272, PEEK (53272) OR8
22Ø FORR=1TO LEN (A$): XT=X
23Ø S=Ø: A=ASC(MID$(A$,R,
    1)): A=A-64: IFA=-32
    THEN X=X+1Ø: XT=X:
    NEXT
24Ø S=S+1: B=L(A,S): S=S+1
25Ø IFB=1Ø THEN 73Ø
3ØØ ONB GO TO 5ØØ,55Ø,57Ø,
    59Ø,61Ø,63Ø,65Ø,67Ø,73Ø
5ØØ FORM=1TOL(A,S)*Y1
51Ø Y=Y-1: GOSUB 1ØØØ:
    NEXT: GO TO 24Ø
55Ø FORM=1TOL(A,S)*Y1
56Ø Y=Y+1: GOSUB 1ØØØ:
    NEXT: GO TO 24Ø
57Ø FORM=1TOL(A,S)*X1
58Ø X=X-1:GOSUB 1ØØØ:
    NEXT: GOTO 24Ø
59Ø FORM=1TOL(A,S)*X1
6ØØ X=X+1:GOSUB 1ØØØ:NEXT
    : GOTO 24Ø
61Ø FORM=1TOL(A,S)*X1
62Ø X=X-1: J=(L(A,S)*Y1)/(L(A,
    S)*X1)
622 JJ=J: IFJ>THEN JJ=J/
    INT(J)
625 FORU=1TOJ:Y=Y-JJ:
    GOSUB 1ØØØ: NEXTU,M:
    GO TO 240
63Ø FORM=1TOL(A,S)*X1
64Ø X=X+1: J=(L(A,S)*Y1)/(L(A,
    S)*X1)
642 JJ=J: IFJ>1THEN JJ=J/
    INT(J)
645 FORU=1TOJ:Y=Y-JJ: GO
    SUB 1ØØØ: NEXT U,M: GO
    TO 24Ø
65Ø FORM=1TOL(A,S)*X1
66Ø X=X-1: J=(L(A,S)*Y1)/(L(A,
    S)*X1)
662 JJ=J:IFJ>1THEN JJ=J/INT
    (J)
665 FORU=1TOJ:Y=Y+JJ:
    GOSUB 1ØØØ: NEXTU,M:
    GO TO 24Ø
67Ø FORM=1TOL(A,S)*X1
68Ø X=X+1: J=(L(A,S)*Y1)/(L(A,
    S)*X1)
682 JJ=J:IFJ>1THENJJ=J/
    INT(J)
685 FORU=1TOJ:Y=Y+JJ:
    GOSUB 1ØØØ: NEXTU, M:
    GO TO 24Ø
73Ø Y=3Ø: X=XT+11*X1:S=Ø
737 NEXT:RESTORE:GOTO 5
74Ø GETA$: IFA$=" " THEN
    74Ø
75Ø POKE 53265, PEEK
    (53265) AND 223: POKE
    53272, 21
76Ø GO TO 157
1ØØØ RO=INT(Y/8): CH=INT(X/
    8): LI=YAND7: BI=7-
    (XAND7):
    BY=8192+RO*32Ø+CH*8
    +LI
1Ø1Ø POKE BY, PEEK (BY) OR
    2↑BI
1Ø2Ø RETURN
3ØØØ DATA 2,8,1,8,4,8,2,8,1,4,
    3,8,1Ø
3Ø1Ø DATA 2,8,4,8,1,2,5,2,3,5,
    4,6,6,2,5,2,3,6,1Ø
3Ø2Ø DATA 4,6,8,2,5,2,3,6,2,8,
    4,6,6,2,1Ø
3Ø3Ø DATA 2,8,4,6,6,2,1,4,5,2,
    3,6,1Ø
3Ø4Ø DATA 4,8,3,8,2,4,4,6,3,6,
    2,4,4,8,1Ø
3Ø5Ø DATA 4,8,3,8,2,4,4,6,3,6,
    2,4,1Ø
3Ø6Ø DATA 4,7,3,7,2,8,4,8,1,4,
    3,2,1Ø
3Ø7Ø DATA 2,8,1,4,4,8,1,4,2,8,
    1Ø
3Ø8Ø DATA 4,8,3,3,2,8,4,4,3,8,
    1Ø
3Ø9Ø DATA 4,8,3,2,2,6,7,2,3,2,
    1Ø
31ØØ DATA 2,8,1,4,4,2,6,4,7,4,
    8,4,1Ø
311Ø DATA 2,8,4,6,1Ø
312Ø DATA 2,8,1,8,8,4,6,4,2,8,
    1Ø
313Ø DATA 2,8,1,8,8,8,1,8,1Ø
314Ø DATA 2,8,4,8,1,8,3,8,1Ø
315Ø DATA 2,8,1,8,4,6,8,2,7,2,
```

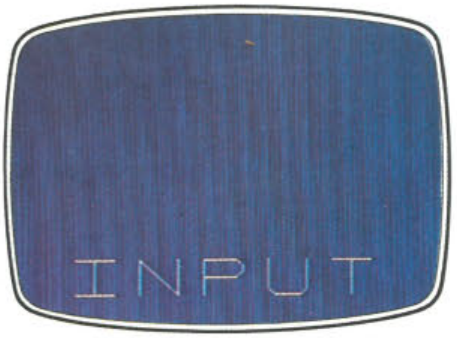



```
    3,6,1Ø
316Ø DATA 2,8,4,8,1,8,3,8,4,8,
    2,8,8,1,5,4,1Ø
318Ø DATA 2,8,1,8,4,6,8,2,7,2,
    3,6,4,3,8,4,1Ø
319Ø DATA 4,8,3,8,2,4,4,8,2,4,
    3,8,1Ø
32ØØ DATA 4,8,3,3,2,8,1Ø
321Ø DATA 2,8,4,8,1,8,1Ø
322Ø DATA 2,4,8,4,6,4,1,4,1Ø
323Ø DATA 2,8,6,4,8,4,1,8,1Ø
324Ø DATA 8,8,5,4,6,4,7,8,1Ø
325Ø DATA 8,4,6,4,7,8,1Ø
326Ø DATA 4,8,7,8,4,7,1Ø
```

Como puedes ver, el programa contiene 26 sentencias DATA, una para cada una de las letras del alfabeto. Y cada una de dichas sentencias contiene una serie de números que indican al ordenador cómo tiene que dibujar la forma de la letra en cuestión, según una serie de trazos cortos. El primer número que figura a continuación de cada DATA es el número total de trazos que se utilizan para formar cada letra. Por ejemplo la L necesita menos que la S. El mayor número utilizado es el 12.

Los siguientes números están ordenados por parejas que dan las coordenadas *x* e *y* de cada uno de los segmentos. Tal como hemos explicado anteriormente, se trata de números relativos y no absolutos, por lo que las líneas que se dibujan realmente pueden ir afectadas por factores de escala. El primer par de coordenadas especifica un punto de partida para la letra dentro de su «caja» imaginaria.

Los números contenidos en las sentencias DATA se leen por medio de las correspondientes READ y se pa-



san a dos *arrays* llamados N y A. A es un *array* tridimensional de 26 × 12 × 2 dimensiones, que corresponden, respectivamente, al número de letras, el número máximo de líneas y los vectores *x* e *y*.

INTRODUCCION DE UNA CADENA DE CARACTERES

Las líneas que te permiten introducir tus propias palabras realizan una comprobación para asegurarse de que en efecto has introducido una cadena de caracteres y no una cadena «nula», pero por lo demás no hay límite en cuanto a la longitud de la cadena a introducir.

Los programas te permiten introducir dos valores: un «factor X» y un «factor Y». Los valores que introduzcas aquí determinan el tamaño real de cada letra (es decir, las dimensiones de la «caja» invisible). Para que te hagas una idea en cuanto a la escala, el número 1 corresponde a un carácter de tamaño estándar. Con el 2 obtendrías doble anchura o doble altura, mientras que el Ø,5 daría como resultado la mitad de altura o anchura. Observa que con valores más pequeños que 1 puedes conseguir algunos efectos inesperados. Como el ordenador no puede dibujar una fracción de pixel, trazará por defecto un doble pixel. En consecuencia, las letras que contengan un gran número de líneas tienen más posibilidades de que les suceda una cosa de este tipo. Como resultado podría ocurrir, por ejemplo, que la S resultara ligeramente superior en tamaño a la T.

Al disponer de valores diferentes para cada uno de los factores, estás en condiciones de producir algunos efectos interesantes para variar los caracteres: ya sea con caracteres altos y estrechos o con caracteres bajos y cortos.

Como no existe un límite para la longitud de la cadena que puedes introducir, tienes que tener cuidado de no teclear demasiadas letras para que el espacio disponible en pantalla para la representación de las mismas resulte suficiente. De no ser así, el ordenador se detendrá y te enviará un mensaje de error o producirá extrañas variaciones en las letras.

Puedes calcular de forma muy sencilla el número de caracteres que te caben para un determinado factor de escala. Observa que el valor que importa es el factor X, ya que es el que define cuánto más ancha o más estrecha es cada letra (el factor Y determina la altura de la letra). Si tienes un valor de 2 como factor de multiplicación, sólo podrás acomodar la mitad de caracteres de los que te cabían en una línea normal. Si en cambio tuvieras un factor de multiplicación de 4, sólo podrías poner la cuarta parte de caracteres.

ANALIZANDO LA CADENA

Al igual que otros programas para crear una determinada tipografía, éste utiliza un bucle principal para recorrer cada letra de la cadena. Dicho bucle empieza en la línea 22Ø.

El programa prosigue de forma análoga a la del anterior generador de letras, definiendo una variable A que hace igual al código ASCII de cada letra. A continuación examina si dicho carácter es una letra de la A a la Z (cualquier otro carácter se trata como si fuera un espacio). De no ocurrir esto, el ordenador ejecuta un NEXT para actualizar el bucle FOR...NEXT, y si ya no hay más letras en la cadena de caracteres el ordenador te permite introducir una nueva cadena.

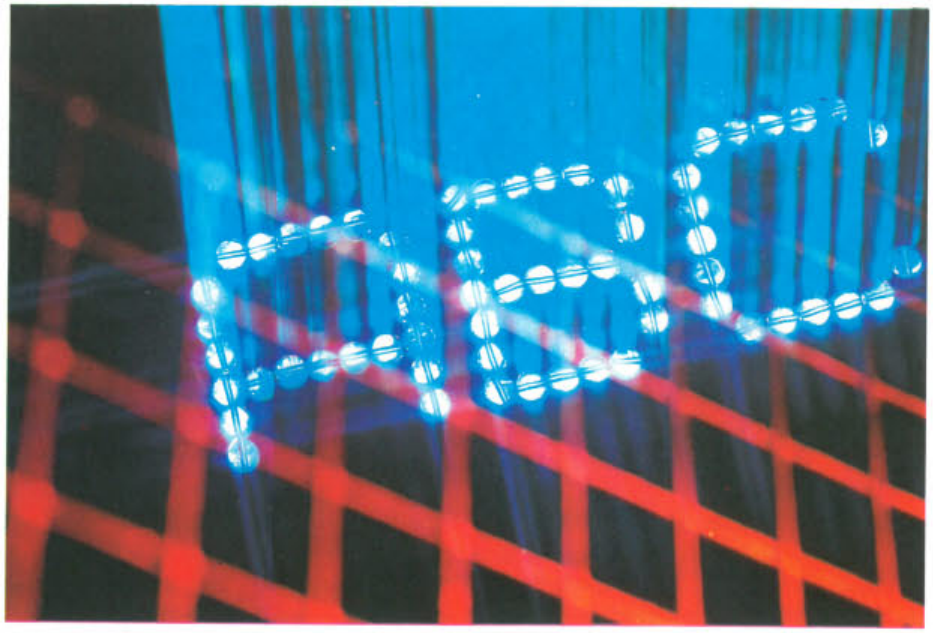
Como el Commodore no puede dibujar líneas en la pantalla con un comando único, tienes que construir las letras a base de puntos pokeados en la pantalla de uno en uno con la subrutina de la línea 1ØØØ. Las rutinas cortas que hay en las líneas 5ØØ a 73Ø determinan la dirección de las líneas que se dibujan; en total hay ocho subrutinas para izquierda, derecha, arriba, abajo y las cuatro diagonales. En las sentencias DATA se determina la longitud de cada línea.

Cuando el ordenador termina de dibujar una letra, pasa a la siguiente letra de la cadena de caracteres, suponiendo que haya una siguiente letra, y repite todo el proceso para dibujarla. Si ya no hay más letras que dibujar, el ordenador regresa a la línea 1ØØ para que puedas introducir una nueva cadena de caracteres.

DISEÑA TUS PROPIOS CARACTERES

Ya conoces tres ejemplos de nuevos caracteres tipográficos, pero puedes utilizar las mismas ideas para crear tus propias letras especiales. Aunque no puedes modificar la doble anchura o altura de los caracteres gráficos propios del ordenador, puedes emplear, sin embargo, las rutinas de «expansión» para ampliar tus propios caracteres redefinidos.

Las variantes que puedes introducir son casi ilimitadas: puedes diseñar tus propios caracteres a partir de los gráficos de tu ordenador y almacenarlos en un *array*. Si no estás satisfecho con los caracteres que lleva incorporados tu ordenador, no hay razón para que no puedas diseñar tus propios GDUs y combinarlos. También podrías construir nuevas variantes de los modelos tipográficos que hemos visto en este artículo; para ello te basta con modificar los valores de los DATA una vez que tengas definidos los nuevos caracteres, y cambiar el *array* en caso de que sea necesario.



USO DE LAS LETRAS GRANDES

Lo de crear letras grandes está muy bien, pero no tendría sentido a menos que puedas incluirlas en tus programas. Si utilizas el programa de dibujo de este artículo, tienes que poder llamar a las letras especiales para poder emplearlas en cualquier parte.

Tienes varias formas posibles para utilizar los caracteres tipográficos especiales en tus propios programas. La solución más evidente consiste en incorporar los programas generadores de letras en forma de subrutina y a continuación poner un GOSUB cada vez que quieras acceder a ella.

Si quieres utilizar la rutina de expansión del conjunto de caracteres, ésta es, probablemente, la mejor solución. Sin embargo, este programa de dibujo de letras que acabamos de ver, dista bastante de ser ideal. La principal razón de esto reside en que ocupa una gran cantidad de espacio de memoria, con lo que no te quedará mucho disponible para tu propio uso.

Puedes solucionar este problema de diversas formas, todas las cuales implican la utilización de programas para dibujar tus propias letras y el almacenamiento de los resultados de forma que queden disponibles para su uso posterior donde haga falta.

Una manera de hacer esto es almacenar tus diseños en GDUs, para lo cual puedes pokear el correspondiente GDU con el contenido de la pantalla.

Otra alternativa es almacenar en cinta con un SAVE la sección de memoria correspondiente a la pantalla como bloque en código máquina. Posteriormente puedes volver a hacer una carga con LOAD del gráfico desde la cinta al ordenador. Mientras dicho gráfico esté en el ordenador, tienes que protegerlo, almacenándolo por encima del extremo del BASIC. Una vez que el gráfico esté en la memoria, ya estás en condiciones de utilizarlo.

La mejor manera de hacerlo consiste en escribir una rutina para extraer mediante PEEKs el correspondiente código contenido en la zona protegida de la RAM y, a continuación, pokear dicho código en la zona de memoria correspondiente a la pantalla, o modificar el conjunto de punteros de caracteres para que tu código «se convierta» en varios caracteres. Seguidamente podrías imprimir los caracteres para hacer que aparezcan tus letras grandes. La rutina para desplazar tu bloque de código desde su área protegida en RAM hasta la pantalla, sería más o menos así:

```
1ØØØ FOR x = 1 TO (longitud de
     tu bloque de código)
1Ø1Ø POKE (dirección inicial de
     la memoria de pantalla) +
     x, PEEK (dirección inicial
     de tu bloque de código) + x
1Ø2Ø NEXT x
```

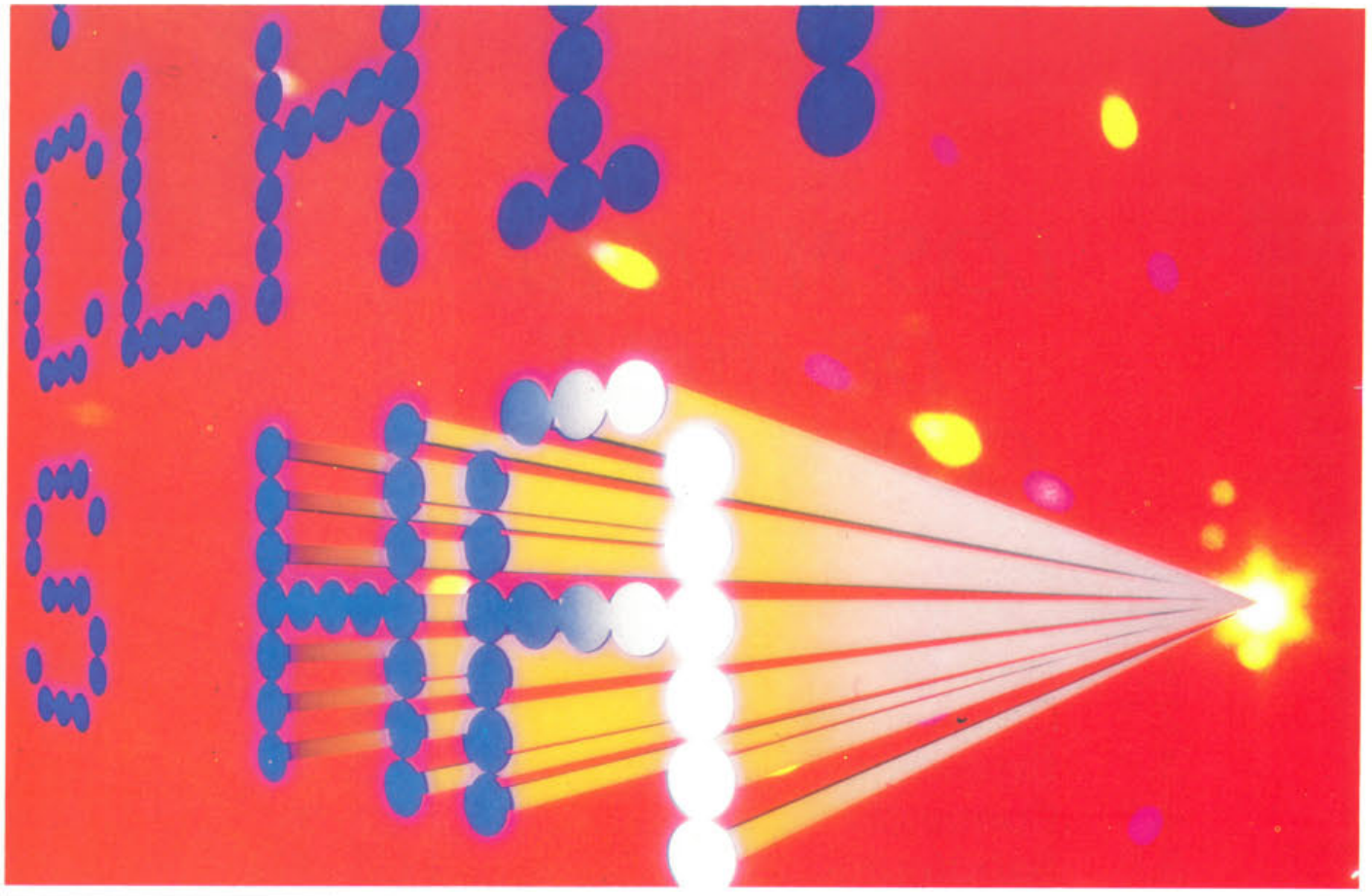
Tampoco en este caso es esta rutina una solución especialmente acertada, ya que tardaría bastante en terminar, a menos que conozcas el código máquina y puedas escribir una rutina equivalente en código máquina. En tal caso podrías llamar a dicha rutina, de forma que tus letras aparecerían en la pantalla de forma casi instantánea.

ALMACENAMIENTO DEL CONTENIDO DE LA PANTALLA

La otra alternativa que tienes es almacenar mediante un SAVE la imagen que tengas en pantalla y cargarla con un LOAD cuando hagas la carga de tu programa. La imagen permanecerá entonces en pantalla hasta que la elimines al hacer un borrado de pantalla.

Aunque los Commodore, a diferencia de otros ordenadores, no pueden cargar fácilmente una imagen de pantalla, sí pueden, en cambio, almacenar en cinta grandes letras. Si no quieres utilizar los actuales programas para crear las letras grandes por medio de subrutinas en tus propios programas, puedes almacenar la rutina en cinta como bloque de memoria y modificar el puntero del conjunto de caracteres para que apunte a tu bloque de memoria y a continuación ya puedes imprimir con las letras grandes mediante una serie de sentencias PRINT.

Los tipos de letras grandes que hemos visto en este artículo no son en absoluto las únicas clases de letras grandes que puedes crear con tu ordenador. Ya has visto cómo una de las formas de almacenar las letras consiste en utilizar los GDUs. También puedes poner varios GDUs juntos, combinándolos para formar una letra más grande. Por medio de este método puedes diseñar GDUs tan complicados como quieras, ya que el ordenador sólo tiene que hacer un PRINT de una cadena de caracteres, no teniendo que dibujar en realidad cada carácter, como hacen los otros programas.



Suscríbase ahora a INPUT commodore

PRECIO DE CUBIERTA PTAS. **375**
MENOS: **20 %** de descuento al suscriptor Ptas. **75**
USTED PAGA SOLO PTAS. **300** (por ejemplar)

SUSCRIPCION ANUAL 12 EJEMPLARES **4.500** Ptas. (**900** Ptas), USTED PAGA SOLO **3.600** Ptas (entrega a domicilio gratis)

20% de descuento

por sólo 300 Ptas. ejemplar, y recibidos todos cómodamente en su hogar

INPUT le proporciona
INFORMACION... DIVERSION... FORMACION...
(un curso completo de programación)...

...LA POSIBILIDAD DE MEJORAR SU NIVEL PROFESIONAL... EL NIVEL DE LOS ESTUDIOS...

...Descubra el mundo de la informática...
...Aprenda a programar con facilidad...
...Diviértase con los ordenadores...
...Esté siempre al día...

Recorte y envíe este cupón de inmediato a EDISA, López de Hoyos, 141 28002 Madrid, o bien llámenos al Telf. (91) 415 97 12

INPUT

BOLETIN DE SUSCRIPCION

SI, envíeme INPUT COMODORE durante 1 año (12 ejemplares), al precio especial de oferta de **3.600 Ptas. AHORRANDOME 900 Ptas.** sobre el precio normal de portada de 12 ejemplares sueltos. (Por favor, cumplimente este boletín con sus datos personales e indíquenos con una (X) la forma de pago por usted elegida, mételo en un sobre y deposítelo en el buzón más próximo).

NOMBRE ..... APELLIDOS .....
DOMICILIO ..... NUM. ..... PISO ..... ESCALERA ..... COD. POSTAL .....
POBLACION ..... PROVINCIA ..... TELF. .....
PROFESION .....

FORMA DE PAGO ELEGIDA: Reembolso ..... ☐ Domiciliación Bancaria ..... ☐
Talón nominativo que adjunto a favor de EDISA ..... ☐

INSTRUCCIONES DE DOMICILIACION BANCARIA (si es elegida por usted)

Muy señores míos: _____ de _____ de 19__
Les ruego que, con cargo a mi cuenta n.º _____ atiendan, hasta nuevo aviso, el pago de los recibos que les presentará Editorial PLANETA-AGOSTINI a nombre de: .....
..... BANCO/C de AHORROS .....
..... DIRECCION .....

FIRMA

LOS MEJORES DE INPUT COMMODORE

PUESTO	TITULO	PORCENTAJE
1.º	Uridium	17,8 %
2.º	Green Beret	13,3 %
3.º	Exploding	13,3 %
4.º	Comando	13,3 %
5.º	Saboteur	11,3 %
6.º	Skyfox	8,9 %
7.º	Rambo	8,9 %
8.º	Hard Ball	4,4 %
9.º	Dragon's Lair	4,4 %
10.º	Alien-8	4,4 %
		100 %

Para la confección de esta relación únicamente se han tenido en cuenta las votaciones enviadas por nuestros lectores de acuerdo con la sección «Los Mejores de Input».

Febrero de 1987

Ayúdanos a ayudarte

Puedes ganar 25.000 pts. o tu juego preferido

Encuesta para conocer tu perfil y preferencias

Hace ahora un año, la redacción de INPUT se inundó de encuestas enviadas por vosotros, los lectores. Gracias a esa entusiasta acogida fue posible conocer a fondo vuestras inquietudes y ajustar a ellas los contenidos de la revista.
Ahora, esta nueva encuesta pretende saber hasta qué punto hemos conseguido ese objetivo, y enterarnos de por dónde van ahora vuestras exigencias, siempre pensando en que INPUT siga siendo una herramienta indispensable para vuestros primeros escarceos por el mundo de la informática.

1. ¿Tienes ordenador? Sí ☐ No ☐
2. En caso afirmativo. ¿Cuál?
   Marca ..... Modelo .....
3. ¿Cuánto tiempo hace que lo posees? .....
4. ¿Qué periféricos y accesorios tienes? .....
5. ¿Desde qué número lees INPUT? .....
6. ¿Con qué frecuencia la adquieres?
   TODAS ☐ DE VEZ EN CUANDO ☐ OTROS ☐
7. ¿Cuántas personas leen tu ejemplar de INPUT?
   Sólo yo ☐ Yo y mis amigos ☐
   Mi hermano, mi padre y yo ☐ Mi hermano y yo ☐
8. ¿Lees toda la revista? Sí ☐ No ☐
9. ¿Tecleas los programas que publicamos?
   Todo ☐ Casi todo ☐
   Alguna cosa ☐ Nada ☐
10. Califica qué secciones te gustan más con mayor puntuación (10 a 0)
    Programación .....
    Código Máquina .....
    Robótica .....
    Coleccionable .....
    Revista Software .....
    Aplicaciones .....
    Actualidad .....
    Libros .....
    Buzón .....
    Educación .....
    Inteligencia Artificial .....
11. ¿Qué secciones o temas echas de menos en INPUT? .....
12. ¿Crees que alguna sección debe ampliarse? ¿Cuál?
    .....
13. ¿Cuál debería reducir su extensión? .....
14. ¿Tienes en cuenta los comentarios sobre juegos (Revta. de Software) a la hora de adquirirlas en el mercado?
    Al 100 % ☐ En gran medida ☐ Poco ☐ Nada ☐
15. ¿De todos los artículos que has leído en INPUT, nos podrías destacar algunos?
    N.º 1 .....
    N.º 2 .....
    N.º 3 .....
    N.º 4 .....
    N.º 5 .....
    N.º 6 .....

SOBRE TI

16. Sexo:
    M ☐ F ☐
    Edad:
17. Hasta 12 años ☐ De 21 a 25 años ☐
    De 12 a 14 años ☐ De 26 a 35 años ☐
    De 15 a 17 años ☐ De 36 a 45 años ☐
    De 18 a 20 años ☐ Más de 45 años ☐
18. ¿Cuál es tu situación?
    Estudio enseñanza básica ☐
    Estudio enseñanza media ☐
    Estudio enseñanza Universitaria ☐
    Estoy parado ☐
    Trabajo (en este último caso responde a la siguiente pregunta) ☐
19. ¿Cuál es tu profesión? .....
20. ¿Con qué objetivo adquiriste tu ordenador?
    Jugar ☐ Educativo ☐ Música ☐
    Utilidades ☐ Profesionales ☐ Dibujo ☐
21. ¿Cuál es tu nivel de conocimientos de informática?
    0 ☐ Bastante ☐ Aficionado con exp. ☐
    Profesional ☐
22. Aparte de la informática, ¿qué otras aficiones tienes?
    Deporte ☐ Cine ☐ Música ☐ ☐
    Vídeo ☐ Pintura ☐ Teatro ☐ ☐
    Motociclismo ☐ Fotografía ☐ ☐ ☐
23. ¿Estás satisfecho con tu ordenador? .....
24. En caso de que te tuvieses que comprar otro, ¿por cuál te decidirías? .....
25. ¿Lees otras revistas de informática? Sí ☐ No ☐
26. ¿Cuáles? .....

Entre los que contestéis a esta encuesta, se sorteará un premio en metálico de 25.000 pts. y diez programas a elegir por los agraciados. Así, que escribe tus datos para participar. La fecha límite es el próximo 5 de marzo. No es preciso que contestes las preguntas para las que no tengas respuesta. Tampoco es necesario que las respuestas sean favorables a nuestra labor para que puedas participar en el sorteo. Critícanos si honestamente crees que debes hacerlo. Envíanos este cuestionario o su fotocopia a: ENCUESTA INPUT COMMODORE, Aribau, 185, 1.º - 08021 Barcelona

Nombre ..... Apellidos .....
Edad .....Dirección ..... Dto. Postal .....
Teléfono de contacto ..... Gracias y suerte con el premio.

LISP, EL LENGUAJE DE LAS LISTAS (I)

Aunque empezó con lentitud, el LISP ha ido ganando poco a poco velocidad y las «máquinas de LISP» pueden muy bien ser los prototipos de los ordenadores domésticos que se vendan en los próximos diez a veinte años.

El LISP fue desarrollado originalmente por John McCarthy en el Instituto Tecnológico de Massachusetts a principios de la década de los 60, una época en que los lenguajes de ordenador más sofisticados eran las versiones primitivas del FORTRAN, el lenguaje parecido al BASIC que utilizan ingenieros y científicos. Su nombre se deriva de List Processing, procesado de listas, lo que nos puede dar una pista sobre la forma en que trabaja. Hoy día, veinticinco años más tarde, los programas en LISP continúan basándose en los mismos principios fundamentales, mientras que el FORTRAN ha debido ir incorporando nuevas capacidades a medida que la gente ha ido aprendiendo gradualmente a escribir mejor sus programas.

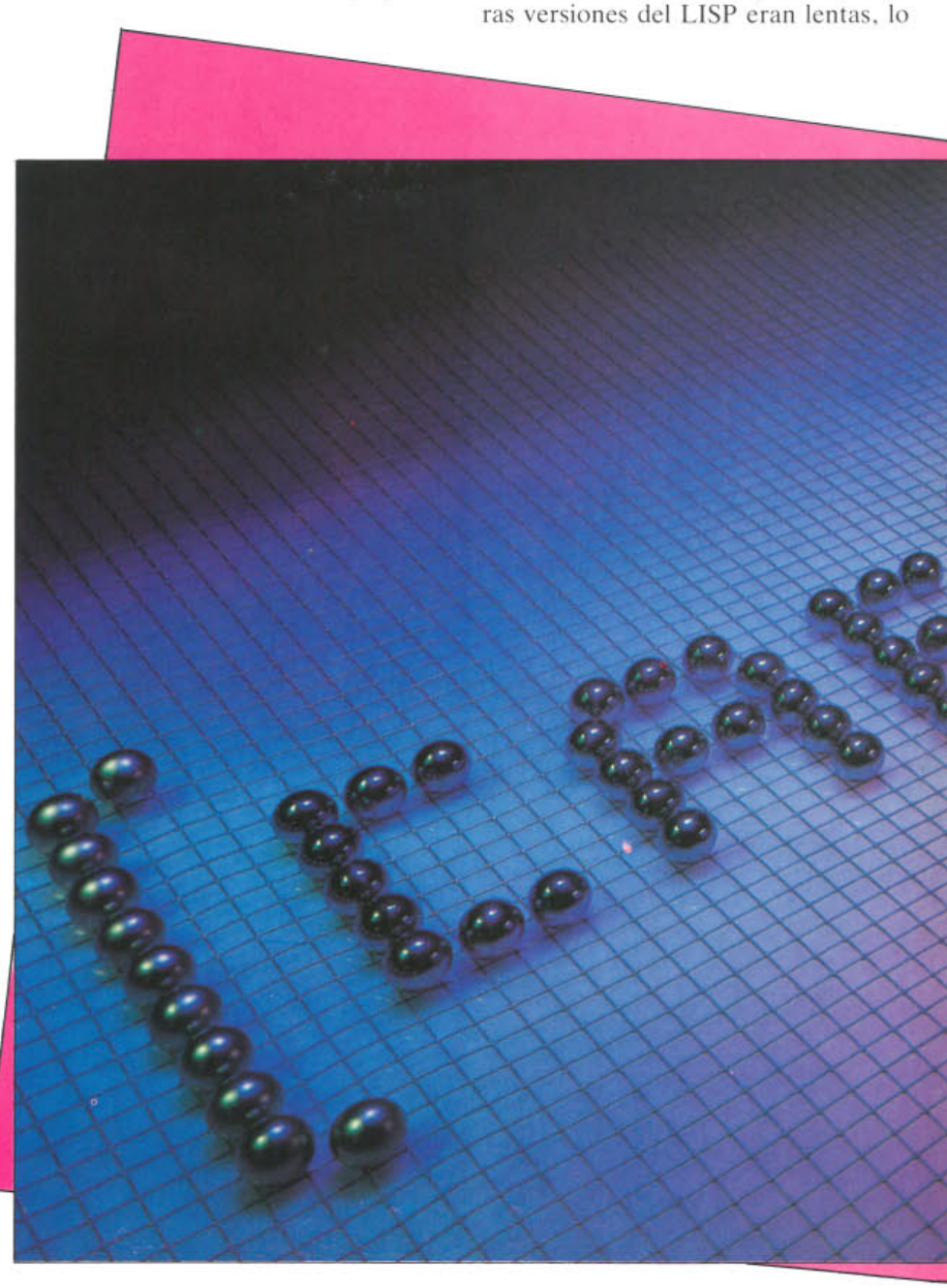
Incluso aunque tú no hayas oído hablar nunca antes del LISP, es muy probable que los programas que escribes se hayan visto influidos por él, debido a que muchos de sus conceptos han sido aplicados a otros lenguajes. No obstante, a pesar de su influencia sobre otros lenguajes, las ideas del LISP siguen siendo radicalmente diferentes de las de lenguajes como el BASIC o el PASCAL, siendo además el LISP un lenguaje considerablemente más potente que ellos.

LA IMPORTANCIA DE LA VELOCIDAD

Es bien conocido que resulta más fácil escribir un programa en BASIC para realizar una determinada tarea, que hacerlo en código máquina. El BASIC es un lenguaje de más alto nivel y sus recursos actúan sobre él como una especie de palanca intelectual haciendo que el programador pueda abordar problemas más difíciles.

En el futuro los ordenadores realizarán tareas mucho más complejas de lo que lo hacen actualmente, lo que significa que los programadores tendrán que utilizar lenguajes aún más potentes, como el LISP, para escribir sus programas; pero los lenguajes más potentes tienden a ser más lentos.

Realmente es cierto que las primeras versiones del LISP eran lentas, lo

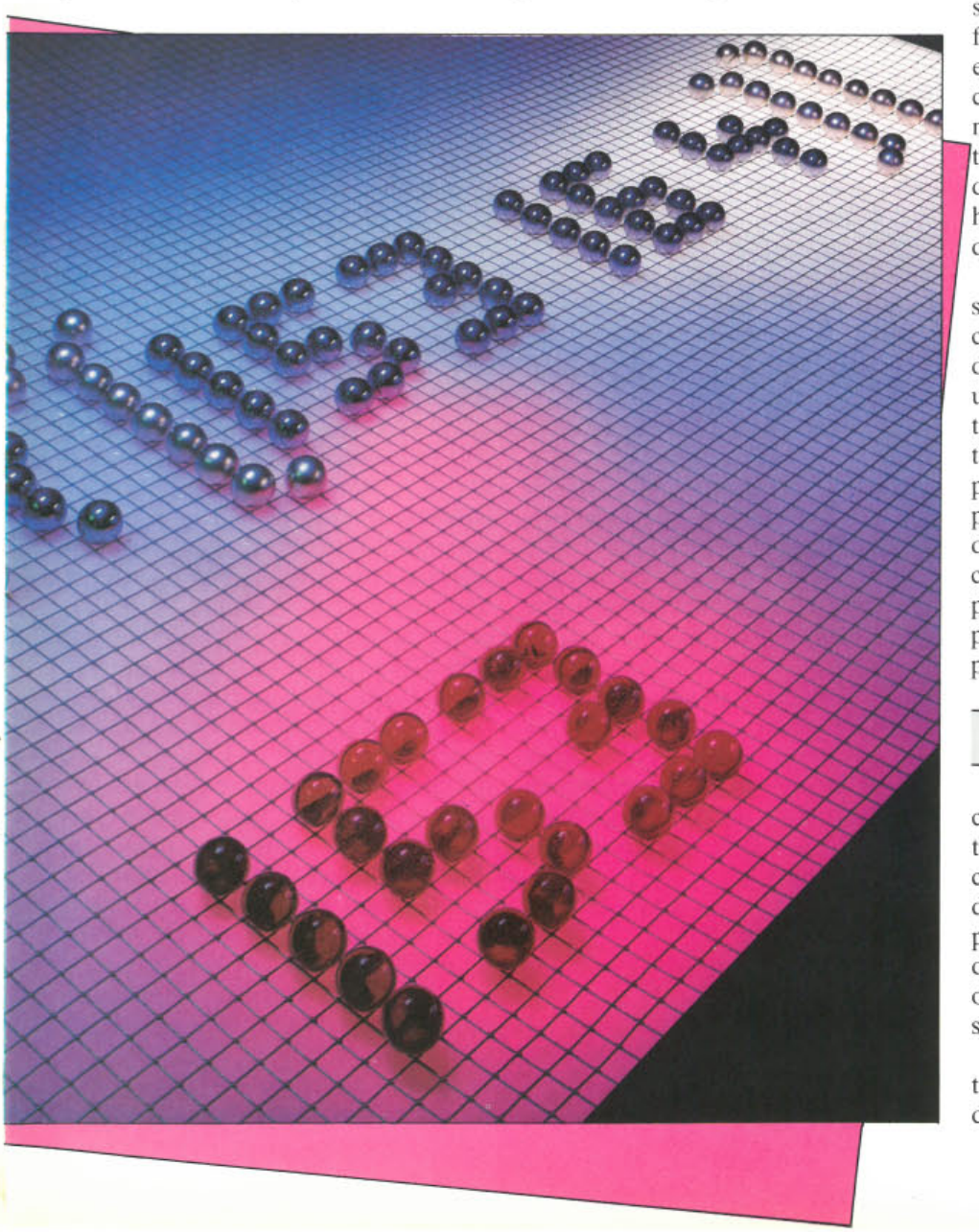


- LA IMPORTANCIA DE LA VELOCIDAD
- INTELIGENCIA ARTIFICIAL
- OTRAS APLICACIONES
- EL LISP Y EL ORDENADOR
- LOS FUNDAMENTOS
- CARGA DEL LISP
- VALORES DE LOS ATOMOS
- LOS NUMEROS EN LISP
- VERDADERO O FALSO
- SEGUIMIENTO

cual le quitó popularidad. Sin embargo, las actuales versiones de LISP que corren sobre grandes ordenadores, son tan rápidas como cualquier otro lenguaje de alto nivel. Además el producto de la revolución tecnológica en los circuitos integrados es un enorme crecimiento de la velocidad y capacidad de procesamiento de un solo chip de silicio. Esto significa que ya no resulta vital apurar hasta el último bit la velocidad del sistema utilizando el código máquina.

INTELIGENCIA ARTIFICIAL

El LISP es el principal lenguaje utilizado en Inteligencia Artificial. Su objetivo es aquí la creación de programas de ordenador que presenten un comportamiento similar en algún aspecto a la inteligencia humana. Un ejemplo típico es la máquina que juega al ajedrez. Otro ejemplo es uno de los más famosos en los que se utilizó el LISP. Se trata de un programa llamado ELIZA que desarrolló el papel de psicoanalista, comunicándose con sus pacientes a través de un terminal de ordenador. El programa simulaba la conversación que habría podido tener lugar, preguntando y respondiendo a las preguntas. Hubo personas que realmente se quedaron confundidas pensando que el ordenador entendía de verdad lo que ellas le decían. De hecho lo que hacía el ordenador era simplemente mirar a su entrada y replicar modificándola adecuadamente. El análisis necesario para hacer esto es mucho más fácil en LISP que en otros lenguajes.



Otra área de la inteligencia artificial son los Sistemas Expertos. En este caso se trata de programar en un ordenador todos los conocimientos de un experto humano en algún campo, tal como el mantenimiento de los motores de automóvil. Un usuario sin experiencia puede averiguar lo que le pasa a su coche tecleando los síntomas de los fallos. El ordenador le replica con su diagnóstico, haciendo nuevas preguntas que pueden ser necesarias para localizar con más precisión el problema.

OTRAS APLICACIONES

Otros programas más tradicionales, como compiladores y procesadores de textos, son mucho más sencillos cuando se escriben en LISP, habiéndose utilizado también este lenguaje para escribir los programas de algunos de los sistemas de diseño asistido por ordenador (CAD) utilizados en el diseño de circuitos integrados.

El LISP ha desempeñado un importante papel en los programas de ordenador que se ocupan del álgebra. Es

evidente que los ordenadores son muy buenos en aritmética. Sin embargo con el LISP es posible hacer que lo sean también en álgebra. Esto significa que los ordenadores pueden hacer la clase de cálculos algebraicos que mantendría ocupado a un ejército de matemáticos y llenaría kilómetros de papel.

Algunos de los cálculos de la física teórica contienen un álgebra tan complicada que sólo puede hacerse mediante estos sistemas basados en el LISP. A un nivel más modesto, una aplicación típica del LISP que podrías hacer correr en tu ordenador doméstico podría ser el cálculo de derivadas, lo cual será muy útil si tienes que hacer docenas y docenas de ejemplos de cálculo en tus tareas de casa.

EL LISP Y EL ORDENADOR

Sería erróneo dar la impresión de que el LISP es el único lenguaje capaz de resolver problemas difíciles o de alcanzar relevancia en el futuro. Han aparecido varios lenguajes modernos que utilizan ampliamente sus conceptos, entre los que se incluyen el LOGO y el PROLOG. Con mucha frecuencia los compiladores e intérpretes de estos lenguajes fueron escritos en LISP. Por eso el LISP es en cierta medida el lenguaje ensamblador de la nueva generación de lenguajes de ordenador. Actualmente existen de hecho procesadores especiales cuyos lenguajes ensambladores están basados en el LISP, básicamente de la misma forma que los procesadores Z80, 6502 o 6809 de un ordenador doméstico son capaces de entender el código máquina adecuado.

Estos procesadores de LISP constituyen muchas veces el corazón de determinados ordenadores personales de alta potencia o de puestos de desarrollo de ingeniería, cuyo principal lenguaje es el LISP. Estas «Máquinas de LISP» pueden muy bien ser los prototipos de los ordenadores domésticos que estén a la venta en la calle dentro de cinco o diez años.

En estos artículos sobre LISP, explicaremos los fundamentos del lenguaje y podrás seguir sus desarrollos incluso aunque todavía no tengas un LISP.

Como en el caso del BASIC, existen diferencias entre las versiones del LISP de ordenadores diferentes, aunque afortunadamente las variaciones de uno a otro suelen ser muy pequeñas; normalmente suelen consistir únicamente en diferencias ortográficas en los nombres de las funciones. Un inconveniente del LISP es que hace uso de un elevado número de paréntesis. Si eres una de esas personas que encuentran grandes dificultades para hacer cuadrar el número de paréntesis en las expresiones del BASIC, puede que el LISP no sea el lenguaje más adecuado para ti.

FUNDAMENTOS

Uno de los hechos que distingue al LISP del BASIC es que el LISP, además de números, es capaz de manipular símbolos. Es algo parecido a la forma en que el BASIC puede fraccionar y manejar cadenas de caracteres, si bien la capacidad de manejo de símbolos del LISP es mucho mayor.

Los objetos básicos que maneja el LISP se llaman átomos y listas. Un átomo es realmente como una variable en BASIC, empieza por una letra y puede continuar por cualquier longitud de letras y números (aunque en la práctica siempre hay un límite). Según esto, podrían ser átomos los siguientes: LUIS, atomo, NUM1, NUM2. Todos estamos familiarizados con muchos tipos de listas, por ejemplo una lista de cosas que comprar o la lista de los jugadores de algún equipo deportivo. Tu lista de compras podría ser: té, azúcar y leche. Una lista LISP es algo muy parecido que incluye cierto número de átomos y probablemente otras listas. Para informar al LISP de que algo es una lista, se encierra ésta entre paréntesis.

Así, tu lista de compras en LISP podría ser algo como:

```
(te azucar leche)
```

Una lista de LISP puede contener tanto átomos como otras listas, por lo que otro ejemplo sería:

```
((te frio) (leche caliente) azucar)
```

La anterior lista contiene dos listas y un átomo. Los paréntesis se utilizan de una manera muy parecida a las dobles comillas que usa el BASIC para mostrar dónde empiezan y terminan las cadenas de caracteres. La diferencia consiste en que las cadenas de caracteres no pueden contener otras cadenas de caracteres.

CARGA DEL LISP

Si quieres hacer tus propias pruebas con el LISP, no tienes necesidad de cambiar tu ordenador doméstico, tan hábil con el BASIC, por una máquina de LISP de seis millones de pesetas. De hecho la mayoría de los ordenadores domésticos se pueden convertir al LISP. La única razón por la que un ordenador entiende el BASIC es porque existe un gran programa en ROM que es un intérprete de BASIC. Es perfectamente posible sustituir este código máquina por otro programa que interprete o entienda los programas en LISP. En la práctica, el añadirle el LISP a algunos ordenadores consiste en enchufarles un chip o un pequeño módulo que contiene una ROM que lleva dentro el código máquina necesario. Se suele adoptar con más frecuencia la solución de mantener en la memoria principal del ordenador la sección de código máquina que contiene el intérprete LISP. En tales casos se carga el LISP a partir de una cinta o disco, igual que cualquier otro programa y una vez que el código está dentro de tu ordenador éste se comporta ya como un sistema LISP.

¿Qué es lo que se vería cuando esto sucede? En vez del acostumbrado ok del BASIC, te encontrarías con el mensaje Evaluate:. Esto significa que el LISP está esperando a que teclees alguna expresión válida en LISP, la cual evaluará a continuación e imprimirá su valor en la pantalla. Así es todo el LISP: tú tecleas expresiones y el ordenador te contesta con sus valores. En LISP se llama expresión a una expresión de tipo simbólico que contenga una lista o un átomo. Con frecuencia suele llamarse a esto una s-

expresion. No existe un comando equivalente al RUN que ponga realmente en marcha el programa. Como verás más adelante, este comando no hace falta.

Cuando te aparece el mensaje Evaluate: y el ordenador se queda esperando, tienes que decidir lo que vas a teclear. Lo más sencillo es un átomo, como LUIS. El LISP te responde entonces con un UNDEFINED. El diálogo sería algo parecido a lo que sigue:

```
Evaluate: LUIS
Value is: UNDEFINED
```

Esto significa que el LISP nunca ha visto anteriormente ese átomo y que no puede encontrar su valor. Al igual que las variables del BASIC, los átomos pueden tener valores. La situación se parece a hacer un PRINT de una variable en BASIC antes de asignarle un valor. Cuando el LISP se encuentra el átomo, intenta evaluarlo, es decir, encontrar su valor. Esta es una importante característica del LISP: todo se evalúa siempre a menos de que exista la intención expresa de no hacerlo.

Hay un método para eliminar las evaluaciones en LISP, la comilla sencilla '. Supongamos que tecleas 'LUIS. En estas condiciones, la respuesta del LISP será ahora LUIS. Con la comilla se detiene cualquier evaluación ulterior de lo que viene detrás. El valor de la expresión LUIS es precisamente el átomo que sigue a la coma. También puede usarse la coma con listas. Así:

```
'(te azucar leche)
```

se evaluará como (te azucar leche)

Una aplicación mucho más interesante que la evaluación de átomos es la evaluación de listas. Nuevamente tenemos un ejemplo de esto en la comilla anterior: 'LUIS es realmente una abreviatura de (QUOTE LUIS). De hecho algunos sistemas sólo aceptarán la segunda forma de la expresión. QUOTE es aquí una función del LISP, mientras que el átomo LUIS es su argumento. Observa que QUOTE no halla el valor de su argumento.

Compara esta función con las funciones del BASIC, que podrían ser del tipo:

```
RND (2Ø)
```

El BASIC encuentra para este valor un número aleatorio. Esto mismo en LISP tendría el siguiente aspecto:

```
(RND 2Ø)
```

Observa la nueva situación de los paréntesis. Análogamente, si en BASIC existiera una función llamada QUOTE, se escribiría así:

```
QUOTE (LUIS)
```

Puede que te confunda al principio este nuevo uso de los paréntesis, pero una vez que se ha entendido es sencillo.

VALOR DE LOS ATOMOS

Al igual que las variables, los átomos también pueden tener un valor. Sin embargo surge la cuestión de cómo adquiere un átomo su valor. Por ejemplo, ¿cómo se le asigna a LUIS el valor 1? Se hace con la función SET de la siguiente forma:

```
(SET' LUIS 1)
```

Como de costumbre, el LISP lo evalúa todo. El primer argumento de SET se evalúa como LUIS, y el segundo se evalúa como 1; a continuación la función asigna el valor 1 a LUIS. Esto nos muestra dos cosas. En primer lugar, que en LISP todos los números son átomos que se evalúan para dar su valor numérico. Y en segundo lugar que, igual que ocurre en BASIC, las funciones pueden tener más de un argumento. Si ahora teclearas LUIS, el LISP te respondería con 1. La interacción con el LISP podría ser algo así:

```
Evaluate: (SET' LUIS 1)
Value is: 1
Evaluate: LUIS
Value is: 1
```

Como todas las demás funciones, ha de resultar un valor para SET, normalmente el valor de su segundo argumento que en este caso es 1. Existe otra versión del SET que es SETQ, en la cual no se evalúa el primer argumento. Así, podrías teclear:

```
(SETQ LUIS 1)
```

sin colocar la comilla delante de LUIS. Para insistir aún más en el tema, supongamos que tecleas seguidamente:

```
(SETQ JORGE LUIS)
```

En este caso, el LISP evaluaría LUIS y encontraría que su valor es 1, y a continuación pondría el valor de JORGE a 1. JORGE no se evalúa y no tiene que ser protegido por medio de '.

La función SET se parece al comando LET que tienen algunas versiones del BASIC. Con frecuencia en BASIC las variables tienen que ser de un tipo declarado. En LISP no hay esta necesidad de especificar las variables: un átomo puede tener como valor un número, otro átomo o una lista. Por ejemplo:

```
(SETQ COMPRAS '(te leche
azucar))
```

asignará tu lista de compras al átomo COMPRAS.

Llegados aquí, puede que te hayas dado cuenta de una importante diferencia entre el LISP y el BASIC. Considera en primer lugar la lista:

```
(te leche azucar)
```

es decir, tu lista de compras. Sin embargo, si el té fuera una función (como SET) esta lista podría ser una expresión en LISP y podría ser evaluada. Más adelante veremos que es fácil definir átomos como té para que sean funciones. La cuestión es que en LISP los programas y los datos tienen exactamente el mismo aspecto. Este hecho tiene gran alcance; significa que un programa en LISP puede manejar otro programa en LISP con la misma facilidad con que maneja otros datos.

En LISP todo se hace por medio de

funciones y encontrando el valor de expresiones. Pero a veces esto no resulta muy conveniente. Por ejemplo, normalmente no estarás interesado en el valor real que resulte de la función SET, es mucho más importante que asigne un valor a un átomo. Por eso algunas funciones sólo se utilizan por sus efectos laterales y no por su valor.

DESMENUZANDO LAS PARTES

Para hacer algo con el LISP, evidentemente tienes que poder manipular listas; es algo análogo al manejo de cadenas de caracteres en el BASIC, que las tres funciones fundamentales que hacían esto son CAR, CDR y CONS. La reacción natural ante esta información es «So what?» (¿Qué?). Suponte sin embargo que te dijeran que estas funciones se llaman FIRST, REST y ADD to FRONT. Estas denominaciones te dan realmente mucha más información sobre lo que realmente hacen. De hecho CAR y CDR se han llamado así por ser las iniciales de «contenido del registro de dirección» (contents of address register) y «contenido del registro de decremento» (contents of decrement register), y proceden de los nombres de dos de los registros del ordenador IBM en el que se utilizó inicialmente el LISP. CONS significa construcción. Casi todo el mundo aprende a utilizar estos términos, pero podrían cambiarse fácilmente por ejemplo por (SETQ FIRST CAR).

Tal como sugiere el nombre propuesto como alternativa, CAR da como valor resultante el primer elemento de la lista. Por ejemplo:

```
(CAR '(te leche azucar))
```

da como valor resultante té. Observa que CAR evalúa sus argumentos, por lo que hay que utilizar una ' para evitar la evaluación de la lista. Para ver de otra forma cómo funciona esto, teclea:

```
(SETQ COMPRAS '(te leche
azucar))
```

y a continuación:

```
(CAR COMPRAS).
```

El LISP te respondería de nuevo con té como valor de esto, ya que se encuentra que el valor de COMPRAS es la lista:

```
(te leche azucar)
```

Con CDR resulta la lista que contiene todos los elementos de la lista dada, tomada como argumento, excepto el primero. Así el valor de:

```
(CDR '(te leche azucar))
```

es la lista:

```
(leche azucar)
```

Tal como sugiere su nombre, CONS sirve para construir listas. Hace esto de una manera muy específica, tiene dos argumentos, un átomo o una lista y una lista. CONS agrega su primer argumento al principio de la lista que figura como segundo argumento. Por ejemplo:

```
(CONS 'galletas '(te leche azucar))
```

se evalúa como:

```
(galletas te leche azucar)
```

Para listas, el funcionamiento es el siguiente:

```
(CONS '(izquierdo derecho) '(mano
pie))
```

que tiene el valor:

```
((izquierdo derecho) mano pie)
```

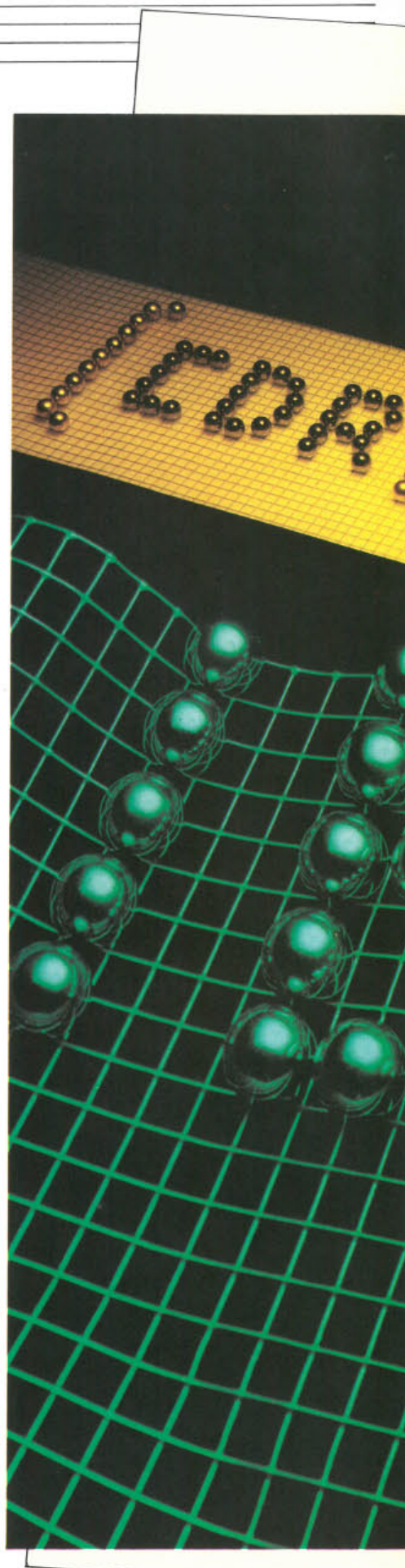
Con CAR y CDR se obtiene en este caso:

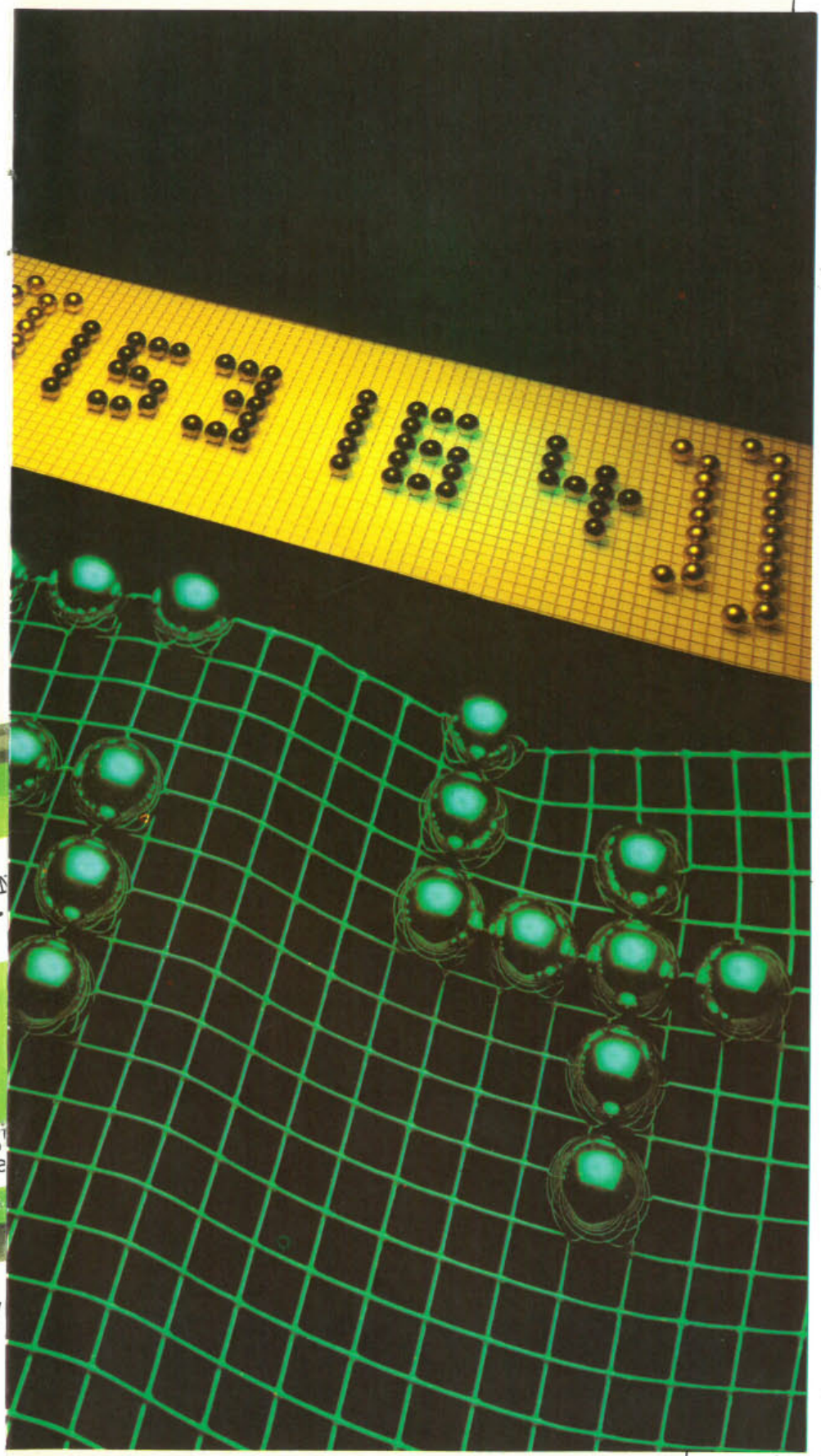
```
(izquierdo derecho)
```

y:

```
(mano pie).
```

Observa que en este caso con CAR se obtiene una lista y no un átomo. Supongamos que tomas el CDR de una lista con sólo un elemento, por ejemplo:





(CDR '(LUIS))

Esto tiene todo el aspecto de una lista vacía (). De hecho en LISP existe un nombre especial para esto, el átomo NIL. Al tomar el CAR o el CDR de un átomo o de una lista vacía se produce un mensaje de error, tal como era de esperar. Existe una generalización de CAR y CDR. Imagínate que tratas de encontrar el átomo izquierdo de la lista:

((izquierdo derecho) mano pie)

Podrías encontrarlo como valor de la expresión:

(CAR (CAR '((izquierdo derecho) mano pie)))

El LISP te permite hacerlo de una forma más corta con la función CAAR, pudiendo escribirse:

(CAAR '((izquierdo derecho) mano pie))

Análogamente, CADR significa tomar el CDR y a continuación el CAR. Los sistemas de LISP permiten también todas las demás combinaciones de CAR y CDR tomados en pares y a veces en ternas, cuaternas y así sucesivamente. Si tuvieras dos átomos LUIS y JUAN y quisieras ponerlos en una lista, podrías sentirte inclinado a teclear:

(CONS 'LUIS 'JUAN)

Aunque se trata de una expresión válida, no es una lista. Su valor es el siguiente par separado por un punto:

(LUIS.JUAN),

pero de momento esto es una complicación innecesaria. Una manera de formar la lista deseada sería teclear lo siguiente:

(CONS 'LUIS (CONS 'JUAN NIL))

El segundo argumento del primer CONS se evalúa para dar la lista (JUAN) en cuyo principio se añade el argumento LUIS. Sin embargo LISP

cuenta también con el método más breve siguiente:

```
(LIST 'LUIS 'JUAN)
```

Este es un ejemplo de otras de las características del LISP: las funciones con un número arbitrario de argumentos. Podrías teclear por ejemplo:

```
(LIST 'UNO 'DOS 'TRES)
```

para obtener la respuesta:

```
(UNO DOS TRES)
```

Esto es muy diferente de lo que ocurre en BASIC, donde las funciones siempre tienen que tener el mismo número de argumentos.

LOS NUMEROS EN LISP

Aunque una de las principales características del LISP es su capacidad de manejar símbolos, también es capaz de manejar números. Las implementaciones del LISP en ordenadores domésticos habitualmente utilizan solamente números enteros. Sin embargo algunos de los grandes sistemas de LISP pueden manejar números en punto flotante y hacer cálculos con ellos con tanta rapidez como cualquier otro lenguaje especial para el cálculo numérico, tal como el FORTRAN. Una peculiaridad que poseen con frecuencia estos sistemas es la aritmética de precisión arbitraria en la que se puede tomar cualquier número de cifras significativas, lo cual resulta perfecto para calcular el valor del número pi con unos cuantos millones de números decimales.

En comparación con el BASIC, el LISP maneja la aritmética de una forma un tanto extraña. Se parece más bien al LOGO, del que ya nos hemos ocupado anteriormente. Los equivalentes de los operadores habituales +, −, *, / y MOD (que da el resto de la división entera) son las funciones PLUS DIFFERENCE, TIMES, QUOTIENT y REMAINDER. La gran diferencia consiste en que el LISP utiliza notación de prefijo, en contraste con el BASIC, que utiliza notación interna. La notación con prefijo se llama a veces notación polaca, en memoria del lógico polaco Lukasiewicz, que demostró que esta notación (así como la polaca inversa, consistente en poner al final el signo de la operación a realizar) son especialmente apropiadas para la aritmética.

Cuando en BASIC quieres sumar 2 y 2, escribes 2 + 2. En LISP esto se convierte en (PLUS 2 2). En la notación con prefijo los operadores aritméticos figuran en primer lugar en las expresiones. Esto permite al LISP tratar a los operadores aritméticos en las funciones exactamente igual que a los demás. Por eso puedes cambiar sus nombres si lo deseas. Algunas de las funciones aritméticas del LISP, como el PLUS, pueden tomar cualquier número de argumentos. Por ejemplo:

```
(PLUS 2 2 2 (TIMES 1 2)
```

se evalúa dando el resultado:

```
2+2+2+1* 2=8
```

En LISP la función MINUS se utiliza para cambiar el signo de un número; por ejemplo el valor de (MINUS 2) es −2. Esto contrasta con el BASIC, donde el símbolo − se emplea tanto para el cambio de signo como para encontrar la diferencia entre dos números.

VERDADERO O FALSO

La siguiente cosa en que tenemos que fijarnos es la forma que tiene el LISP de manejar las expresiones lógicas y cómo hace los saltos condicionales dependiendo del valor de una expresión lógica. En BASIC los valores lógicos se tratan exactamente igual que la aritmética entera; típicamente se utiliza 0 para falso y − para verdadero, aunque son posibles otras elecciones. Una expresión lógica es algo del tipo:

```
A > B
```

Si A es mayor que B, entonces esta expresión en BASIC tiene valor verdadero, en caso contrario tiene valor falso. En LISP verdadero y falso se representan por dos átomos especiales T y NIL (el mismo NIL descrito anteriormente como la lista vacía).

Los valores verdadero y falso resultan de funciones especiales a las que con frecuencia se les llama predicados. Un ejemplo de ello es la función ATOM, que da como resultado el valor T si su argumento es un átomo y NIL en caso contrario. Tecleando:

```
(ATOM 'LUIS)
```

el LISP responderá con el resultado T, mientras que:

```
(ATOM '(leche galletas))
```

dará como resultado NIL. Otros predicados de LISP son GREATERP, para ver cuál de dos átomos tiene mayor valor numérico y NULL que examina una lista para ver si está vacía. El predicado EQ ensaya la igualdad de dos átomos cualesquiera:

```
(EQ'LUIS'COME) (EQ 'LUIS'LUIS)
```

da como resultado T, mientras que:

```
(EQ'LUIS'ENRIQUE)
```

tiene el resultado NIL. En realidad EQ ensaya si sus dos argumentos tienen el mismo valor, así con:

```
(SETQ F '(mermelada pasteles)
```

se tendrá que:

```
(SETQ G F)
```

seguido de:

```
(EQ GF)
```

Tendrá el valor T puesto que los valores resultantes de G y F son la misma lista. Además de éstos, hay muchos otros predicados en LISP. Además de los operadores aritméticos, el LISP cuenta también con el usual complemento de los operadores lógicos. Estos también utilizan la notación de prefijo. Por ejemplo, (OR T T) tiene el valor T, (OR T NIL) tiene el valor T, (AND T NIL) tiene el valor NIL, etc.

En BASIC lo primero que suele hacerse una vez que se está en condiciones de evaluar una expresión lógica es realizar saltos condicionales dependiendo de los valores de dichas expresiones, lo cual se hace con la sentencia IF. El equivalente de esto en LISP se hace por medio de la función COND. Esta función tiene un número arbitrario de argumentos (llamados cláusulas), cada uno de los cuales es una lista que contiene un cierto número de s-expresiones (listas de átomos). Cuando se evalúa la función COND se examinan todas las cláusulas por turno evaluando la primera s-expresión. Si su valor resulta ser T, se examinan todas las s-expresiones subsecuentes de esa cláusula. El valor de la función COND se toma entonces como el de la última s-expresión de la cláusula. En caso contrario la evaluación continúa en la siguiente cláusula. Si ninguna de las cláusulas tiene una primera s-expresión que sea T, la función COND tiene entonces el valor NIL. Normalmente se organizan las cosas de modo que ningún valor NIL para una primera s-expresión dé el mismo resultado que un valor T.

Un caso típico de COND sería el siguiente:

```
(COND  ((EQ LUIS
        1)(PRINT'UNO))
        ((EQ LUIS
        2)(PRINT'DOS))
        ((EQ LUIS
        3)(PRINT'TRES))
```

En este ejemplo se ha introducido otra nueva función, PRINT; su efecto es imprimir el resultado después de evaluarlo. Aunque el LISP cuenta con un conjunto completo de estas funciones de impresión, suelen diferir de un sistema a otro.

Si ahora LUIS ha sido puesto a 2 con SET y se impone la anterior COND, imprimirá la palabra DOS en la pantalla. De hecho el valor de la función PRINT es usualmente el valor de su argumento, por lo que el LISP devolverá el valor DOS como valor de la COND. Está claro por qué sucede esto. En primer lugar el LISP compara el átomo LUIS con 1 y se encuentra con que no son iguales; esto significa que la primera s-expresión de la primera cláusula es NIL. Entonces el LISP se desplaza a la segunda cláusula de la COND. Esta vez la primera s-expresión tiene el valor T, por lo que se evalúa la segunda s-expresión (PRINT 'DOS) con el resultado de imprimir DOS en la pantalla y devolver el valor de la s-expresión (PRINT 'DOS), es decir, DOS.

Todo esto puede parecer bastante complicado y además hay que estar muy atento a los paréntesis en estas expresiones. Sin embargo hay una sentencia muy parecida en el BASIC que hará más claro el funcionamiento de COND. Examina el siguiente fragmento de programa en BASIC:

```
IF LUIS=1 THEN PRINT «UNO»
ELSE IF LUIS=2 THEN PRINT
«DOS»
ELSE IF LUIS=3 THEN PRINT
«TRES»
```

Esta cadena de IF THEN ELSE es equivalente a la función COND. En BASIC estas cadenas terminan muchas veces con un simple ELSE. Esto significa que si no se satisface ninguno de los IFs el último ELSE dominará sobre todos los otros casos. Para tener este comportamiento en LISP, la función COND se termina con la cláusula:

```
(T (código en LISP para otros
casos))
```

Si COND se encuentra con que no se satisface ninguno de sus otros argumentos, llega a esta cláusula y evalúa T que tiene el valor T. Así, las expresiones que siguen a T siempre son evaluadas.

SEGUIMIENTO

Puede que te resulte difícil recordar todo lo que sigue. El LISP te facilita las cosas por medio de una lista de objetos. Dicha lista contiene todos los átomos que el LISP sabe que tienen un valor. La función OBLIST o algo parecido tiene como valor la lista de objetos. Así, para ver todos los objetos conocidos por el LISP, teclea:

```
(OBLIST)
```

Fíjate que aunque esta función no tiene parámetros, también tiene que ir entre paréntesis. En la lista de objetos encontrarás todos los nombres de las funciones familiares como CAR y CDR. También, si haces algo como lo siguiente:

```
(SETQ LUIS 1)
```

te encontrarás el átomo LUIS añadido a la lista.

El hecho clave que permite escribir programas en LISP es, como en el LOGO, que el usuario puede definir sus propias funciones. Estas figurarán entonces en la lista de objetos y, en lo que al LISP respecta, son tan importantes como las que lleva incorporadas el propio LISP. En consecuencia puedes ampliar el LISP para que incluya cualquier cualidad peculiar que quisieras que tuviera; en la segunda parte de esta serie desvelaremos los misterios de cómo usar y definir funciones en LISP.

NO OLVIDES EL TELEFONO...

Cuando, por cualquier motivo, nos escribas, no olvides indicar tu número de teléfono. Así nos será más fácil y rápido ponernos en contacto contigo. Gracias.

INPUT commodore

SERVICIO DE EJEMPLARES ATRASADOS

¡NO TE PIERDAS NI UN SOLO EJEMPLAR!

INPUT COMMODORE quiere proporcionar a sus lectores este nuevo servicio de ejemplares atrasados para que no pierdan la oportunidad de tener en sus hogares todos los ejemplares de esta revista, líder en el mercado español.

Podréis solicitar cualquier número de **INPUT COMMODORE** que querais, siempre al precio de cubierta (sin más gastos).

Utiliza el cupón adjunto, enviándolo a **EDISA** (Dpto. de Suscripciones), López de Hoyos, 141 - 28002 Madrid, o bien llámanos por teléfono al (91) 415 97 12.

CUPON DE PEDIDO

SI, envíenme contrarreembolso ..... ejemplares de **INPUT COMMODORE** de los números:

(marca con una (X) tu elección)

1	2	3	4	5	6	7	8	9	10	11	12	13

NOMBRE

APELLIDOS

DOMICILIO

NUM. PISO ESCALERA COD. POSTAL

POBLACION PROV.

TELEFONO FIRMA

siempre a tu servicio

```
3500 COLOUR1:PRINTTAB(14,
     F)"G□O□L□D":FORZ=Ø
     TO25ØSTEP1Ø:SOUND1,-
     15,Z,1:NEXT:Z=INKEY
     (15Ø)
3550 A(M,5)=A(M,5)+1:A(M,
     3)=A(M,3)+C(M,3):A(M,
     1)=A(M,1)+(A(M,
     3)*ER):A(M,
     6)=Ø:R(M)=Ø:GOTO33ØØ
```

Puedes acceder a esta rutina desde dos sitios del programa. Como ya has podido ver, se te ofrece la opción de iniciar las excavaciones desde la rutina de Exploración e Informe. Pero también se utiliza cuando optas por aumentar la profundidad de la mina en 200 metros, eligiendo el número 3 de la lista de opciones.

Como de costumbre, la primera línea de la rutina se limita a borrar la pantalla y a definir sus colores. La línea 3010 comprueba que se ha completado la fase de exploración para que pueda iniciarse la fase de excavaciones. La línea 3020 vuelve a ocuparse de nuevo de la pantalla, dejándola lista para nuevas presentaciones.

Las líneas 3022 y 3090 se ocupan de los gráficos que muestran en pantalla la mina de oro. La línea 3100 ilustra la excavación y genera algunos efectos sonoros.

La línea 3120 examina si la excavación ha llegado al nivel donde se encuentra el oro y si hay oro en la mina (podría ser que se llegara al nivel esperado para el oro y ocurriese que la mina no contiene absolutamente nada). Si se llega hasta el oro, el programa salta a la línea 3500, que informa al jugador de que se ha encontrado, haciendo sonar una pequeña melodía. La línea 3550 ajusta ahora el valor de los activos del jugador, con arreglo al valor del oro encontrado.

Si no hay oro, el programa continúa hasta la línea 3130. Si la excavación ha sobrepasado la altura esperada para el oro en 200 metros, el jugador es informado de que la mina no lo contiene. Si la excavación aún no ha llegado tan lejos, el jugador recibe inmediatamente el mensaje NO HAY ORO TODAVIA.

Teclea para Commodore-64

```
4ØØØ POKE 5328Ø,7:POKE
     53281,7:PRINT
     "[SHIFT+CLR/
     HOME][CTRL+7]"
4Ø2Ø PRINT "[ 6 ESPACIOS][
     2*CRSR ABAJO][CTRL+9]
     AGENCIA DE CAMBIO "
4Ø3Ø PRINT "[ 4*CRSR
     ABAJO]EL CAMBIO
     ACTUAL ES:-"
4Ø4Ø PRINT :PRINT "[ 3*CRSR
     DCHA.]1KG DE ORO=$"ER
4Ø5Ø PRINT :PRINT :PRINT
     "ENTRA EL N.DE KG A
     CAMBIAR":INPUT NT
4Ø6Ø IF NT>A(M,3) THENPRINT
     "[CTRL+9]TU NO TIENES
     TANTO ORO!!!"
4Ø7Ø NT=INT (NT)
4Ø9Ø IF NT>A(M,3) OR NT<Ø
     THEN4Ø2Ø
41ØØ A(M,3)=A(M,3) - NT:A(M,
     2)=A(M,2) + (NT *
     ER):A(M,1)=A(M,1) + (NT
     * ER)
413Ø PRINT "[CTRL+5]"NT"KG
     CAMBIADOS POR":PRINT
     "[CRSR DCHA.]$"NT *
     ER:FOR F=1 TO
     2ØØØ:NEXT :RETURN
```

Teclea para Vic-20

```
4ØØØ PRINT"[SHIFT+CLR/HOM
     E][CTRL+7]
4Ø2Ø PRINT"[CRSRABAJO][CR
     SRABAJO][CTRL+9]□□□
     AGENCIADECAMBIO□□□
     □"
4Ø3Ø PRINT"[CRSR
     ABAJO][CRSR ABAJO] EL
     TIPO DE CAMBIO □□
     ACTUAL ES:-"
4Ø4Ø PRINT:PRINT"1 KG=$"ER
4Ø5Ø PRINT:PRINT"[CRSR
     ABAJO] ¿CUANTOS KG. DE
     ORO QUIERES □□□□
     CAMBIAR?":INPUT
4Ø6Ø IFNT>A(M,
     3)THENPRINT"[CRSR
     ABAJO]NO HAY TANTO
     ORO □!□!□!□□"
4Ø7Ø NT=INT(NT)
4Ø9Ø IFNTE>A(M,
     3)ORNTE<ØTHEN4Ø2Ø
41ØØ A(M,3)=A(M,3)—NT:A(M,
     2)=A(M,2)+(NT*ER):A(M,
     1)=A(M,1)+(NT*ER)
413Ø PRINT"[CTRL+5]"NT"KG
     CAMBIADOS POR
     $":PRINT"[CRSR IZQUIER
     DA][CTRL+9]'NT*ER:FOR
     F=1TO2ØØØ:NEXT:
     RETURN
```

La línea 4000 configura el programa.

La línea 4020 presenta el título en la pantalla, el valor de mercado del oro y los mensajes para seleccionar el número de kilogramos que hay que vender. La línea 4070 comprueba si posees suficiente oro. La línea 4080 sirve para asegurarse de que la cantidad de oro vendido es un número entero.

La línea 4090 vuelve a enviar el programa al punto de presentación de mensajes si la cantidad que se pretende vender supera a la cantidad de oro que se posee, o es menor que cero. La línea 4100 modifica el valor de los activos totales con arreglo a la cantidad de oro vendido.

Esta subrutina informa al jugador de cuánto oro se ha vendido y cuántos dólares se han recibido a cambio, cosa que se hace en la línea 4130. La línea 5000 corresponde a la opción de pasar sin hacer nada.

LOS TOQUES FINALES

Teclea para Commodore-64

```
7ØØØ POKE 5328Ø,3:POKE
     53281,3:PRINT
     "[SHIFT+CLR/
     HOME][CTRL+1]"
7Ø1Ø PRINT"[9*CRSRABAJO]"
     TAB(16);A$(M):PRINTTA
     B(12)"[CTRL+9][CTRL+
     3]TEHASARRUINADO!!!"
7Ø15 FOR Z=1 TO 1ØØØ:NEXT
7Ø3Ø POKE 198,Ø:WAIT198,
     1:RUN5
```

```
60000 POKE 56334,0:POKE 1,
      35
60010 FOR F=0 TO 2047:POKE
      12298 + F,PEEK(53248
      + F):NEXT
60020 POKE 1,39:POKE 56334,
      1
60030 SP=12808
60040 READ A:IF A= -1
      THEN60070
60050 POKE SP,A
60060 SP=SP + 1 GOTO 60040
60070 RETURN
60080 DATA 255,85,170,0,0,0,
      0,62,28,56,126,28,62,
      120,28
60090 DATA 255,255,62,126,
      127,60,124,126,0,0,0,
      0,1,1,1,1
60100 DATA 7,29,49,45,255,
      255,91,126,128,96,48,
      80,152,140,252,138
60110 DATA 1,1,1,49,49,49,
      49,255,122,187,62,95,
      153,255,153,126
60120 DATA 209,177,224,128,
      128,128,128,0,0,128,
      128,64,32,32,16
60130 DATA 1,3,7,7,4,4,7,7,
      255,255,255,255,149,
      149,159,159
60140 DATA 24,126,153,255,
      126,153,126,219,128,
      192,224,240,240,248,
      168,248,255
60150 DATA 16,8,8,4,14,31,
      31,255,-1
```

Teclea para Vic-20

```
70000 PRINT"[SHIFT+CLR/
      HOME]"
70010 PRINT"[CRSRABAJO]"TA
      B(6);A$(M):PRINT'[CRSR
      IZQUIERDA][CTRL+9][CT
      RL+3]HASHECHOBANCA
      RROTA!!!"
70015 FOR Z=1TO1000:NEXT
70020 PRINT"[CTRL+7]PULSA
      UNA TECLA PARA JUGAR
      OTRA VEZ"
70030 POKE198,0:WAIT198,
      1:RUN
```

Las líneas 7000 a 7030 contienen una rutina para jugar otra vez el juego.

Las líneas comprendidas entre 8000 a 8070 contienen los DATA para los GDUs.

¡Ahora ya puedes amasar tu inmesa fortuna y comprar todas esas cosas que siempre te has prometido!

ADIVINA MIS PALABRAS

- UN JUEGO PARA DOS JUGADORES
- PREPARACION DE LA PANTALLA
- LAS REGLAS DEL JUEGO
- VALORES DE LAS LETRAS
- ESTRATEGIA

El juego de las palabras de INPUT es adecuado para todas las edades, puedes hacerlo tan difícil o tan fácil como desees, y es increíblemente adictivo

Los juegos de ordenador no tienen por qué ser únicamente recreativos como los juegos para salón o como algunos de simulación domésticos, sino que también pueden ser docentes.

El Ahorcado es el conocido juego que puede adaptarse para que corra en un ordenador. Este juego puede ayudar a tus amigos a mejorar su expresión, ampliar sus conocimientos generales, a dominar más el idioma, etc. Elige un tema como el de la ingeniería química y pronto dominarás una gran parte de su terminología.

El Juego de Palabras de **INPUT** es de este estilo, es para dos jugadores y se basa en adivinar palabras o frases. Es más interesante y divertido que **El Ahorcado**, y es igual de educativo. Puedes jugarlo como **El Ahorcado**, limitado a un determinado tema, puedes utilizar palabras con un determinado número de letras, emplear citas de Cervantes o lo que más te guste.

EL JUEGO

Primero podrás introducir los nombres de los dos jugadores. Después tienes la opción de elegir el número de palabras de las frases que introduce cada jugador. Una interesante faceta de este juego es que cuanto más largas son las frases, algunas veces, son más fáciles de adivinar, porque mayor es el número de pistas-prueba.

Una vez establecido el número de palabras, tendrás que elegir el número de vueltas que deberá efectuar el programa.

A continuación, el primer jugador deberá pensar en una frase e introducirla. No es necesario que el oponente esté encerrado y gritando en un armario mientras se efectúa la introducción, puesto que las letras no aparecen al pulsarlas. Pero si tienes un oponente honesto, tienes la opción de que las letras vayan apareciendo en la pantalla al pulsarlas. Si las letras pueden verse en la pantalla, se evita una escritura errónea de la frase, eliminando así las discusiones cuando aparece.

Entre cada palabra de una frase sólo debe haber un espacio. La longitud máxima de cualquier frase es de 77.

Una vez completada la frase, hay que pulsar la tecla de entrada y aparece la pantalla principal. En la parte superior de la misma se sitúa la puntuación de cada jugador. Al principio del juego, cada jugador tiene 200 puntos, y el total puede aumentar o disminuir a medida que progresa el juego.

Debajo de las puntuaciones hay una tabla con los valores de las letras: las más corrientes tienen valores altos, y las menos corrientes, valores bajos. La frase misteriosa se presenta en forma de una fila de asteriscos.

En el fondo de la pantalla aparece un juego de instrucciones y un espacio para que introduzcas tus comandos y tus adivinanzas.

ESTRATEGIA

El jugador tiene tres opciones: comprar letras, adivinar una letra en una posición específica, o adivinar la frase entera.

En las primeras etapas del juego, una buena elección es comprar un espacio, aunque hay que asegurarse de

que la frase contiene más de una palabra. La forma de proceder a partir de ahí es asunto tuyo. Las vocales son caras, pero tienen una probabilidad muy elevada de que sean válidas, mientras que las letras más baratas son arriesgadas debido a su rareza. A veces, las palabras son más fáciles de adivinar si se han encontrado algunas consonantes —un derroche de vocales no siempre es de ayuda.

A medida que la frase va tomando forma, probablemente verás que puedes adivinar una letra de una determinada posición. Por ejemplo, puedes tener una palabra como Q·E. Es evidente que la letra central es una U. De esta manera puedes sumar puntos. Una letra elegida correctamente sumará su valor a tu puntuación, mientras que una errónea sólo restará la mitad de su valor. Teclea XX para seleccionar la opción de adivinanza e introduce la tuya empleando el cursor.

Cuando haya varias letras en su lugar, es posible que tengas un destello de inspiración y desees adivinar toda la frase. Para ello, teclea ZZ y la podrás introducir. Si la frase es correcta, el valor de las letras que faltaban se sumarán a la correspondiente puntuación. En cambio, si es errónea, se restarán 50 puntos. Un exceso de adivinanzas a tontas y a locas pronto dejará a cero tu puntuación.

Ahora introduce la primera sección del Juego de Palabras. Estas líneas dejan la pantalla preparada para empezar el juego. Si haces correr el programa con RUN, verás que funciona, pero no podrás ir más allá, puesto que el resto del programa que cubre las distintas elecciones se termina en el siguiente capítulo.

No olvides salvar esta parte del programa con un buen SAVE.

```
 5 POKE 5328Ø,1:POKE 53281,
   1:PRINT "[SHIFT+CLR/
   HOME][CTRL+7]":CHR$(8)
 6 SP$=" ":FOR z=1 TO
   4Ø:SP$=" " + SP$:NEXT Z
1Ø R$="PALABRA":
   W=14:F=1:TA=2ØØ:TB=2ØØ
16 GD$="[CLR/HOME]":FOR
   Z=1 TO 23:GD$=GD$ +
   "[CRSR ABAJO]":NEXT Z
4Ø A$="JACK":PRINT
   "[SHIFT+CLR/
   HOME]NOMBRE JUGADOR
   1":INPUT A$:A$=LEFT$(A$,
   11)
5Ø B$="JILL":PRINT
   "[SHIFT+CLR/
   HOME]NOMBRE JUGADOR
   2":INPUT B$:B$=LEFT$(B$,
   11)
7Ø PRINT "[SHIFT+CLR/
   HOME]ENTRAR NIVEL DE
   DIFICULTAD":PRINT
   "(NUMERO DE PALABRAS EN
   LA FRASE 1-9)"
9Ø GET C$:C=VAL(C$):IF C<1
   OR C>9 THEN 9Ø
11Ø PRINT "[SHIFT+CLR/
   HOME]ENTRAR NUMERO
   DE VUELTAS 1-9"
13Ø GET T$:T=VAL(T$):IF T<1
   OR T>9 THEN 13Ø
15Ø IF C>1 THEN
   J$="S":I$="CON UN SOLO
   ESPACIO ENTRE
   ELLAS":R$="FRASE"
155 PRINT "[SHIFT+CLR/
   HOME]"
16Ø PRINT A$;",ES TU
   TURNO":PRINT "[CRSR
   ABAJO]ENTRA TU FRASE
   DE"; C;
162 PRINT "PALABRA";
   J$:PRINT "SI DESEAS VER
   LAS LETRAS QUE ENTRAS"
165 PRINT "PULSA EL 'Ø', SI
   NO PULSA UN '1'
   PARA": PRINT "SEGUIR...
   [CRSR ABAJO]"
17Ø GET K$:IF K$=" " THEN
   17Ø
19Ø IF K$="1" THEN PRINT
   "?[CTRL+2]":INPUT
   S$:PRINT
   "[CTRL+7]":GOTO 22Ø
2ØØ IF K$="Ø" THEN INPUT
   A$:GOTO 22Ø
21Ø GOTO 17Ø
22Ø I=LEN(S$):PRINT
23Ø IF I=Ø THEN PRINT
   "ENTRADA ILEGAL-[ 7
   ESPACIOS]REPITE !":
   GOSUB 95Ø:GOTO 155
24Ø IF I>64 THEN PRINT
   "ENTRADA DEMASIADO
   LARGA-[5 ESPACIOS]
   REPITE !":GOSUB
   95Ø:GOTO 155
25Ø FOR N=1 TO I=IF MID$(S$,
   N,1)=CHR$(32) THEN D=D
   + 1:GOTO 27Ø
26Ø IF MID$(S$,N,1)<"A" OR
   MID$(S$,N,1)>"Z" THEN
   265
263 GOTO 27Ø
265 PRINT "CARACTER ILEGAL-[
   5 ESPACIOS]REPITE !":
   GOSUB 95Ø:GOTO 155
27Ø IF C=1 AND D=1 THEN
   275
273 GOTO 28Ø
275 PRINT "NO SE PERMITEN
   ESPACIOS EN UNA SOLA ":
   PRINT "PALABRA-[ 4
   ESPACIOS]REPITE !"
277 GOSUB 95Ø:D=Ø:GOTO 155
28Ø NEXT N
29Ø IF D<>C - 1 THEN 295
293 GOTO 3ØØ
295 PRINT "SE SUPONE QUE
   DEBES ENTRAR"; C; "[ 2
   ESPACIOS]PALABRAS "; I$;
   " —REPITE! "
297 GOSUB 95Ø:D=Ø:GOTO 155
3ØØ Z$=" "
31Ø FOR N=1 TO 1:Z$=Z$ +
   "*":NEXT N
321 PRINT "[SHIFT+CLR/
   HOME][CRSR
   ABAJO]PUNTUACION DE
   "A$; TAB(2Ø);
   "PUNTUACION DE"B$
325 PRINT SP$;"[CRSR
   ARRIBA]"TA; TAB(2Ø);TB
33Ø PRINT
   "[CTRL+1][CTRL+9][ 8
   ESPACIOS]VALORES DE
   LOS CARACTERES[ 7
   ESPACIOS][CTRL+Ø] [CTRL
   + 7]";
34Ø FOR N=Ø TO 26:READ
   G$:G$=G$ + G$ +
   "[SHIFT+B]":NEXT N:PRINT
```

```
    G$:RESTORE:GOSUB 2ØØØ
35Ø PRINT LEFT$(GD$,
    1Ø)"[COMM+3]LA "; RS"
    CONTIENE"; I;
    "LETRAS[COMM+7]":PRINT
    Z$
36Ø PRINT LEFT$(GD$,
    17)"[CRSR
    DCHA.][CTRL+6]XX =
    ADIVINAR LETRA,
    ZZ=ADIVINAR FRASE"
37Ø PRINT TAB(3)"UN
    ESPACIO= COMPRAS ESTE
    CARACTER.[CTRL+7]":D$="
    ":PRINT
    "[CTRL+1][CTRL+9]"SP$
    "[CRSR IZQDA.][CTRL+7]"
9ØØ DATA A-2Ø,B-1Ø,C-1Ø,D-12,
    E-2Ø,F-Ø8,G-12,H-Ø8
91Ø DATA I-2Ø,J-Ø4,K-Ø6,I-1Ø-
    M-1Ø,N-1Ø,O-2Ø-P-1Ø,Q-Ø2,
    R-12,S-12
92Ø DATA T-12,U-2Ø,V-Ø8,W-
    Ø8,X-Ø4,Y-Ø8,Z-Ø2,
    "[COMM+ +]-2Ø"
```

La línea 5 establece los colores de la pantalla. La línea 1Ø inicializa todas las cadenas y las variables necesarias en el juego. Esta línea es notablemente corta, porque no hay necesidad de inicializar variables a cero ni inicializar cadenas nulas.

Las líneas 4Ø a 7Ø son las indicaciones del principio del juego. Las líneas 4Ø y 5Ø son los nombres de los jugadores. El número de palabras de la frase se introduce en la línea 7Ø.

La siguiente rutina, línea 9Ø, es una serie de comprobaciones de validez, que aseguran que el número de palabras de la frase está dentro de los límites del programa.

El número de palabras de la frase se llama C$ y la línea 9Ø comprueba si la entrada sólo tiene un carácter de longitud. La línea 9Ø comprueba si la entrada es de 1 a 9, y entre las dos líneas se comprueba que la entrada esté entre el 1 y el 9 y sea un solo número. La línea 1ØØ convierte la cadena en una variable numérica.

Las líneas 11Ø a 13Ø corresponden al número de vueltas elegido. La línea 11Ø es la indicación y pide el número de vueltas, T$. La línea 12Ø es una comprobación de validez similar a las anteriores.

La línea 15Ø comprueba que el número de palabras de la frase sea mayor que uno, e indica a los jugadores que debe haber un espacio entre las palabras. R$ se hace igual a "FRASE", empleado posteriormente en las indicaciones.

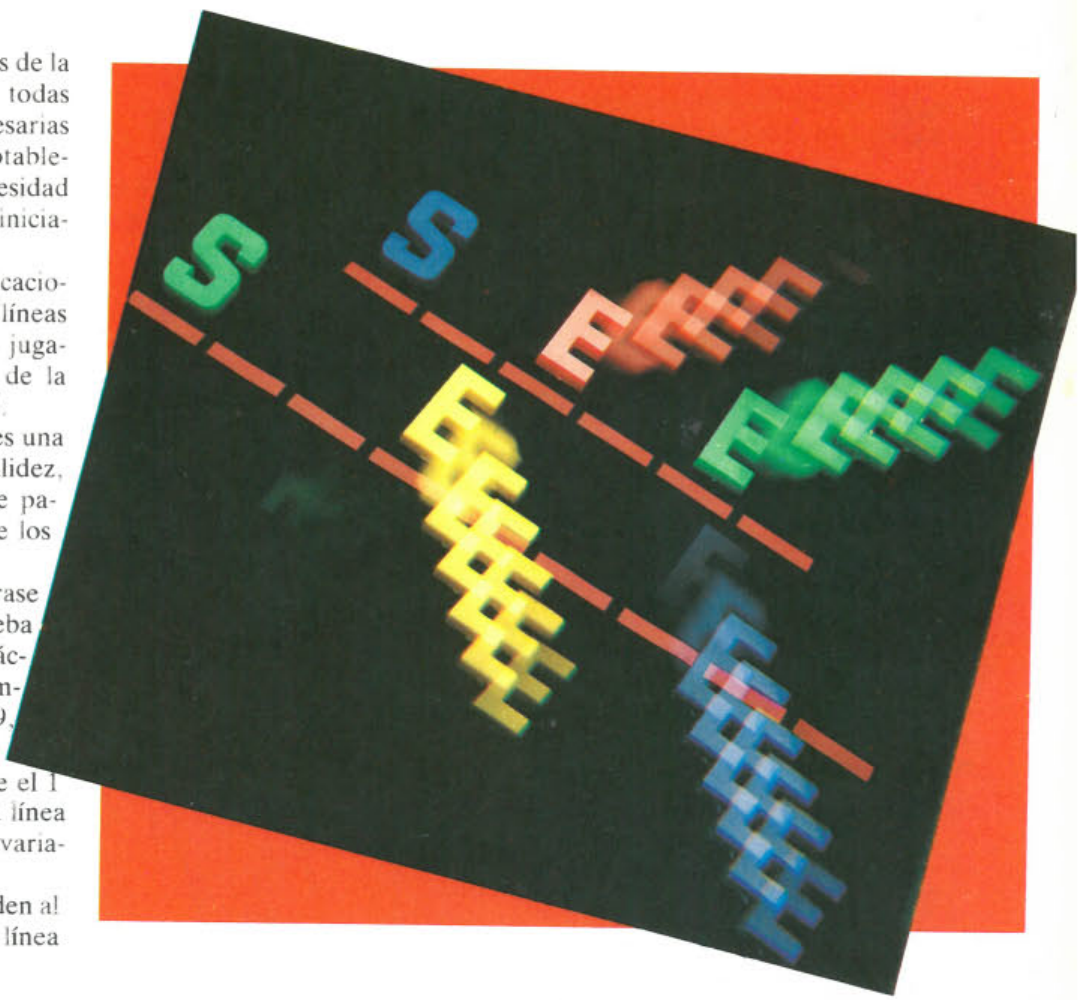
A continuación, el programa introduce la rutina de entrada, que se extiende de la línea 16Ø a la 22Ø, y da instrucciones al jugador que tiene el turno para teclear la frase misteriosa. La selección de un Ø hará que la frase aparezca en la pantalla tal y como fue introducida; en caso contrario, permanece totalmente invisible. La frase se llama S$.

Las líneas 23Ø a 29Ø son comprobaciones de validez. Si la longitud de la frase es menor que un carácter, o sea si el jugador sólo ha pulsado RETURN o ENTER, la línea 23Ø anuncia que se ha producido una entrada ilegal. La línea 24Ø, si la entrada tiene la longitud correcta, y la línea 25Ø comprueba el número de espacios (que debe ser inferior en uno al número de palabras dado por C). Las líneas 26Ø y 27Ø comprueban, junto con la 29Ø, si hay de alguna forma entradas ilegales.

Las líneas 3ØØ y 31Ø establecen Z$, una cadena ficticia que consiste en una cantidad de asteriscos igual a la longitud de S$.

La rutina final, entre las líneas 321 a la 36Ø, establece el resto de la pantalla principal.

Lee (READ) una tabla de valores de letras de la DATA del final del programa, imprimiendo las dos puntuaciones de los jugadores y presentando la cadena ficticia de los asteriscos. La línea 36Ø es una indicación para el jugador que tiene la ocasión para adivinar.



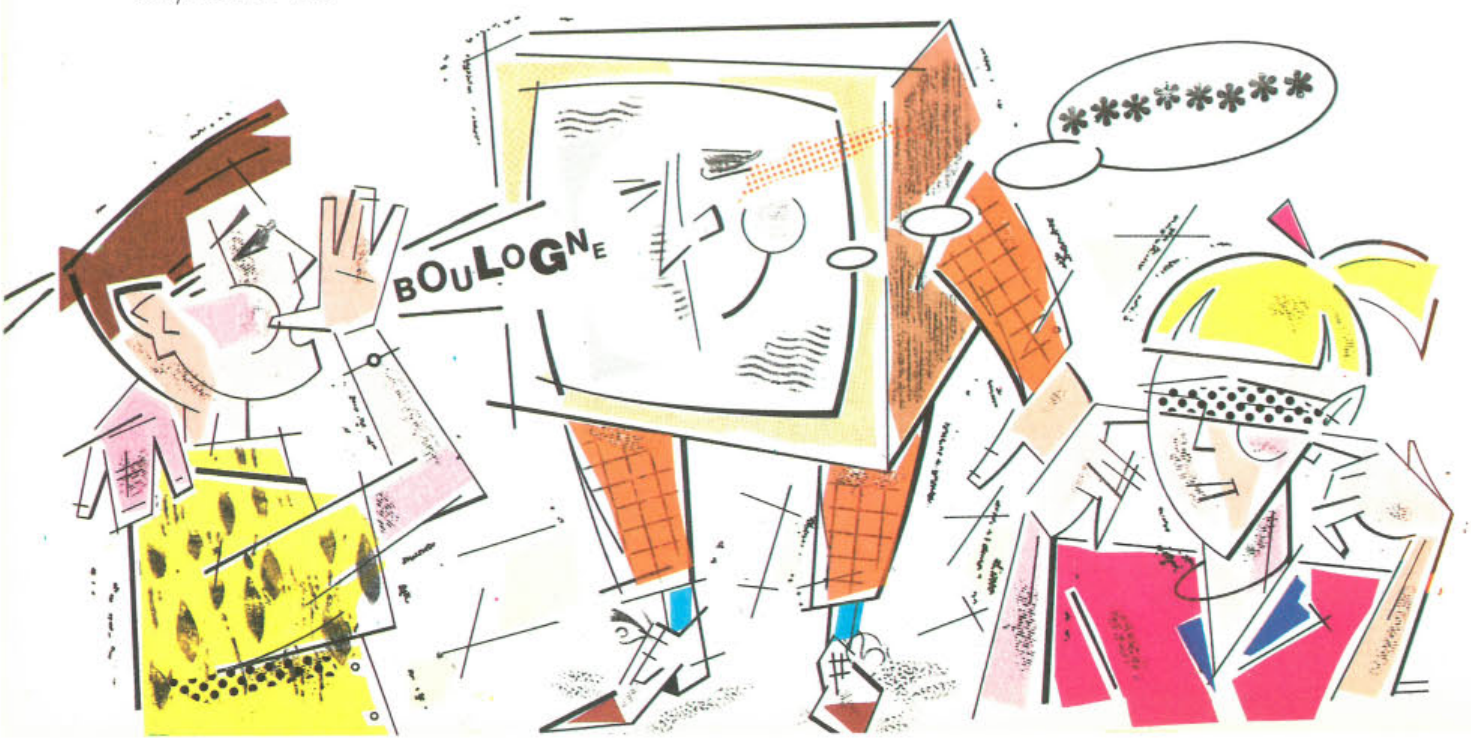
ULTIMANDO EL JUEGO DE LAS PALABRAS

Saca quinario, xileno y quincuagésimo de los polvorientos rincones de tu diccionario. Utiliza tácticas sucias **para sacar de quicio a tus amigos con el juego de las palabras**

En esta segunda parte final sobre el Juego de las Palabras de INPUT hay todo lo que necesitas para empezar a jugar. Teclea el resto de líneas para ver algunas interesantes aplicaciones del manejo de cadenas con tu máquina. Después trata de desconcertar a tus amigos con oscuras palabras y frases.

En el programa hay unas rutinas para cada una de las tres opciones: comprar letras, adivinar una letra específica de una posición específica y adivinar la frase completa. El programa también sigue las puntuaciones, el número de intentos y el número de vueltas.

```
375 INPUT D$
38Ø IF D$=" " THEN
    M=131:D$" ":GOSUB
    81Ø:GOTO 415
385 IF D$="ZZ" THEN D$=" ":
    GOTO 1ØØØ
39Ø IF D$="XX" THEN D$=" ":
    GOTO 5ØØ
4ØØ IF D$<"A" OR D$>"Z" OR
    LEN(D$)>1 THEN GOSUB
    2ØØØ:GOTO 36Ø
41Ø GOSUB 79Ø
415 IF G=Ø THEN 36Ø
42Ø E=Ø
43Ø E=E + 1
44Ø IF E=1 + 1 THEN TB=TB -
    9:G$=LEFT$(G$,M - 1) +
    "[ 4 ESPACIOS]" +
    MID$(G$,M + 4)
445 IF E=1 + 1 THEN PRINT
    "[CLR/HOME][ 5*CRSR
    ABAJO]"G$:D$=" ":GOTO
    47Ø
45Ø IF MID$(S$,E,1)><=D$
    THEN 43Ø
46Ø IF MID$(S$,E,1)=D$ THEN
    Z$=LEFT$(Z$,E - 1) + D$
    + MID$(Z$,E + 1):GOTO
    43Ø
47Ø GOSUB 95Ø:GOSUB 2ØØØ
48Ø PRINT "[CLR/HOME]
    [2*CRSR ABAJO]"TAB(2Ø);
    TB"[CRSR IZQDA.][ 2
    ESPACIOS]":PRINT
    "[COMM+7]"LEFT$(GD$,
    11); Z$
485 PRINT LEFT$(GD$,
    13)SPC(15)"[CTRL+3][CTRL
    + 9]ADIVINA"; F:F=F +
    1:IF S$=Z$ THEN 73Ø
49Ø GOTO 36Ø
5ØØ GOSUB 2ØØØ:PRINT
    LEFT$(GD$,21);:INPUT
    "ADIVINA QUE CARACTER";
    D$
51Ø IF LEN(D$)>1 THEN 5ØØ
52Ø IF D$=CHR$(32) THEN
    GOSUB 79Ø:GOTO 55Ø
53Ø IF D$<"A" OR D$>"Z"
    THEN 5ØØ
54Ø GOSUB 79Ø
55Ø PRINT "[CTRL+4]EN QUE
    POSICION[ 2 ESPACIOS]-[ 2
```



- JUEGO DE PALABRAS DE INPUT COMPLETO
- CON NUEVAS RUTINAS
- COMPRA DE LETRAS
- COMPROBACION DE LA ENTRADA DEL JUGADOR
- COLOCACION DE UNA LETRA ESPECIFICA
- EN POSICION
- ACERTAR LA FRASE
- ¡TRAMPA!

```
    ESPACIOS]USA EL CURSOR
    G 'Ø' PARA ENTRARLO."
56Ø GOTO 66Ø
57Ø GET G$:IF G$=" " THEN
    57Ø
59Ø IF G$="[CRSR DCHA.]"
    AND B<1-1 THEN B=B+1
6ØØ IF G$="[CRSR IZQDA.]"
    AND B>Ø THEN B=B - 1
61Ø IF G$="Ø"THEN 68Ø
66Ø PRINT"[COMM+7]"LEFT$
    (GD$,11); Z$:PRINT"[CRSR
    ARRIBA]"; TAB(B);
    "[CTRL+9][CTRL+3]*[CTRL
    +7][CTRL+Ø]"
67Ø GOTO 57Ø
68Ø IF MID$(Z$,B + 1,1) <>D$
    THEN 69Ø
685 TB=TB-G:PRINT"[CLR/
    HOME][CTRL+9][CTRL+
    5]"TAB(17)"TRAMPA
    [CTRL+7]":FORDE=1TO
    1ØØØ:NEXT:B=Ø:GOTO47Ø
69Ø IF MID$(S$,B + 1,1)=D$
    THEN 7ØØ
695 TB=TB-9/2:PRINT"[CLR/
    HOME][CTRL+9][CTRL+5]
    "TAB(15)"MALASUERTE
    ![CTRL+7]":FORDE=1TO
    1ØØ:NEXT:B=Ø:GOTO47Ø
7ØØ IF MID$(S$,B + 1,1)<>D$
    THEN 71Ø
7Ø5 PRINT"[CLR/HOME][CTRL+
    9][CTRL+5]"TAB(15)
    "BUENACIERTO[CTRL+
    7]":FORDE=1TO1ØØ:NEXT
7Ø6 Z$=LEFT$(Z$,B) + D$ +
    MID$(Z$,B + 2)
7Ø8 TB=TB + G:B=Ø
71Ø IF S$=Z$ THEN 73Ø
72Ø GOTO 47Ø
73Ø PRINT"[CLR/HOME][CTRL+
    9][CTRL+5]
    ENHORABUENA,"; B$;
    :GOSUB95Ø:PRINT"[SHIFT
    +CLR/HOME][CTRL+7]"
74Ø K=K + 1:IF K=T * 2 THEN
    88Ø
75Ø C$=A$:A$=B$:B$=C$
76Ø TC=TA:TA=TB:TB=TC
77Ø G$=" ":D=Ø:F=1
78Ø GOTO 16Ø
79Ø M=(ASC(D$) - 64) * 5 - 4
81Ø IF MID$(G$,M,4)="[ 4
    ESPACIOS]" THEN
    G=Ø:GOTO 83Ø
82Ø G=VAL(MID$(G$,M + 2,2))
83Ø RETURN
88Ø IF TA>TB THEN PRINT
    "[SHIFT+CLR/HOME]"A$;"
    HAS GANADO POR "; TA;
    "PUNTOS":PRINT "A";TB
89Ø IF TA=TB THEN PRINT
    "[SHIFT+CLR/HOME]EL
    RESULTADO ES UN
    EMPATE"
895 END
95Ø FOR DE=1 TO 15ØØ:NEXT
    DE:RETURN
1ØØØ GOSUB 2ØØØ:PRINT
    LEFT$(GD$,
    2Ø)"[CTRL+3]ENTRA LA
    FRASE -[CTRL+7]":INPUT
    GU$
1ØØ5 IF GU$=S$ THEN 1Ø2Ø
1Ø1Ø PRINT "[CLR/HOME][CTRL
    +9][CTRL+5]"TAB(17)
    "ERRONEA![CTRL+7]";
    TB=TB-5Ø:PRINT"[CLR/
    HOME][2*CRSRABAJO]
    "TAB(2Ø);TB"[CRSR
    IZQDA.][2ESPACIOS]
    ":GOSUB95Ø
1Ø15 GOSUB2ØØØ:PRINTLEFT$
    (QD$,13)SPC(15)"[CTRL+
    3][CTRL+9]ADIVINA";
    F:F=F+1:GOTO36Ø
1Ø2Ø FOR N=1 TO
    I:D$=MID$(Z$,N,1):IF
    D$<>"*" THEN 1Ø5Ø
1Ø3Ø M=(ASC(MID$(S$,N,1)) -
    64) * 5 - 4:IF M= - 164
    THEN M=2Ø9
1Ø4Ø TB=TB + VAL(MID$(Q$,M
    + 2,2))
1Ø5Ø NEXT N:PRINT "[CLR/
    HOME][ 2*CRSR
    ABAJO]"TAB(2Ø);TB:GOTO
    73Ø
2ØØØ PRINT LEFT$(QD$,2Ø);
    SP$;SP$;SP$;SP$;SP$;"
    [CLR/HOME][CTRL+9][CTRL
    +3]";SP$;"[CRSR IZQDA.]
    [CTRL+Ø][CTRL+7]":
    RETURN
```

COMPRA DE LETRAS

Si un jugador decide comprar una letra, lo primero que hace el ordenador es comprobar su valor. Para ello, el programa va a la subrutina que empieza en la línea 79Ø.

Ésta comprueba el valor ASCII de la letra y después realiza un cálculo de este valor para determinar el punto de la tabla de letras en que está. Si el ordenador encuentra un blanco, significa que la letra ya ha sido vendida, por lo que el programa vuelve a la línea 36Ø para que el jugador vuelva a hacer una selección. En caso contrario, retira la parte de la cadena que contiene el valor, que fija utilizando VAL. Ésta se utilizará para actualizar la puntuación del jugador.

La rutina que viene después de la compra empieza en la línea 43Ø. Las líneas 43Ø a 46Ø examinan la frase, buscando la existencia en ella de la palabra comprada. La cadena ficticia se actualiza, sustituyendo los asteriscos por la letra en cada posición que aparece en la frase, para visualizar a con-

tinuación la nueva cadena. Si la letra está en la frase, su valor se suma a la puntuación del jugador y, si no está, se resta la mitad de su valor. También se incrementa el total de adivinanzas, F. La línea 44Ø sustituye la letra elegida y su valor por blancos para indicar que esta opción ya no está disponible.

LETRAS ESPECIFICAS

Si el jugador que adivina desea probar una letra en una posición específica, deberá seleccionar XX. Esto hace que el programa salte a la línea 5ØØ y, a continuación, el jugador deberá introducir la letra elegida. Hay varias trampas de error para asegurar que la entrada sea legal. Si se hace una elección legal, el jugador recibe instrucciones sobre el posicionado del cursor para introducir la letra. Con el cursor situado en la posición adecuada, hay que pulsar la tecla Ø. El programa se asegura de que la elección está disponible y comprueba si ésta es correcta.

Si la letra es errónea, la línea 69Ø dice MALA SUERTE al jugador y resta la mitad del valor de la puntuación del jugador como multa. Si la letra y la posición son correctas, la línea 7ØØ dice BUEN ACIERTO al jugador y suma el valor de la letra a su puntuación. Si la frase ha quedado completa, la línea 71Ø envía el programa a la línea 73Ø, que presenta ENHORABUENA al jugador.

LA FRASE ENTERA

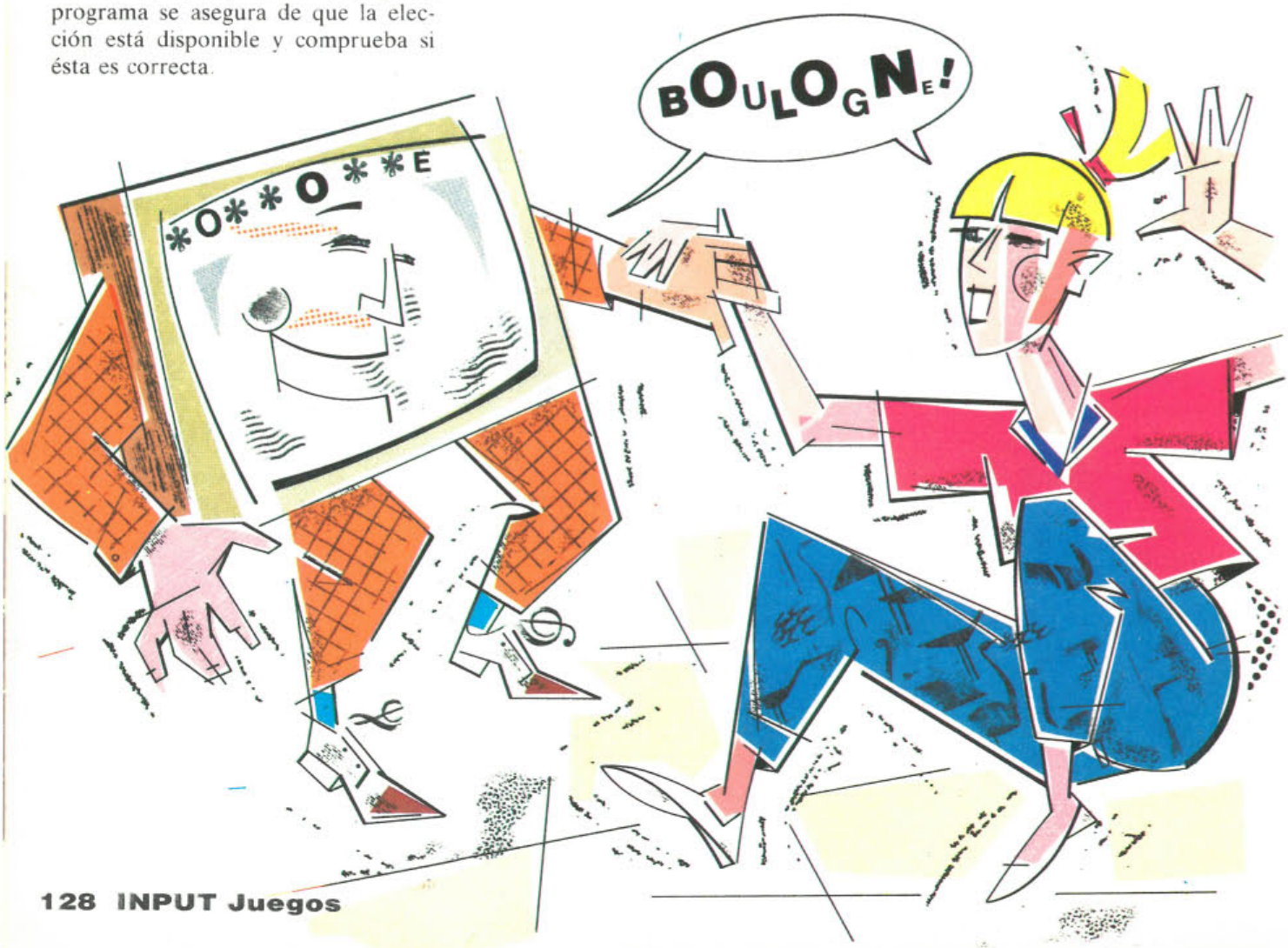
Si el jugador que adivina es más ambicioso y ha elegido la opción de adivinar toda la frase (ZZ), la línea 385 envía el programa a la línea 1ØØØ. La rutina pide al jugador que introduzca la frase, la cual es comparada con la que se introdujo inicialmente. Si la frase es errónea, se restan 5Ø puntos de la puntuación del jugador y se incrementa la cuenta de adivinanzas. Si es correcta, se calcula la puntuación con las letras no utilizadas, y el programa vuelve a la línea 73Ø, que presenta ENHORABUENA.

EL FINAL

Una vez adivina correctamente la frase, el programa comprueba si se ha utilizado el número de tiradas de cada jugador que se ha elegido al principio del programa. Si no ha sido así, el turno pasa al siguiente jugador, una vez puestas a cero todas las cadenas y las variables utilizadas en el juego.

Si el juego ha terminado, el programa salta a la línea 88Ø. En ella y en las dos líneas siguientes se comparan las puntuaciones, y se visualiza el resultado del juego.

El programa termina aquí, pero si lo deseas, puedes añadir una rutina "¿otro juego?" para hacer que el programa y el juego queden completos.



APRENDE MECANOGRAFIA POR ORDENADOR

- ¿POR QUE APRENDER A ESCRIBIR A MAQUINA?
- COMO UTILIZAR EL PROGRAMA
- MEJORANDO LA VELOCIDAD Y LA PRECISIÓN

Tanto si te dispones a sentarte para empezar a escribir tu primera novela, como si lo único que pretendes es tardar menos en teclear tus programas, el primer paso esencial es la capacidad de escribir a máquina con rapidez y precisión.

La quinta generación de ordenadores nos promete una comunicación mucho más directa entre máquina y usuario, considerando como corrientes cosas tales como el control de la máquina por la voz. Algunos de estos nuevos sistemas de control ya se han abierto camino en el campo de las máquinas industriales y comerciales de bajo costo.

Sin embargo, para los entusiastas del micro doméstico, parece que por desgracia, aún tendrá que pasar algún tiempo hasta que estas novedades estén disponibles, por lo que de momento seguimos estando encadenados al teclado.

En realidad el teclado es un método razonablemente eficaz de comunicarle instrucciones e información al ordenador; su principal desventaja consiste en que su uso requiere mucho tiempo, en especial si no eres un mecanógrafo experto (y muy preciso). Y no hay cosa más frustrante que intentar copiar una lista impresa teniendo que estar constantemente cambiando la vista desde el teclado a la pantalla y de ésta al papel, para ir comprobando que lo que vas tecleando es correcto.

En consecuencia, ¿por qué no utilizar tu ordenador para que mejore tu capacidad de escribir a máquina? Se trata de una habilidad que no sólo resulta útil para escribir programas, sino que también resultará de gran valor en cuanto desees hacer uso de toda la potencia de tu ordenador como procesador de textos.

EL PROGRAMA

Exactamente igual que se hace en los cursos de mecanografía que pueden llegar a costar una pequeña fortuna, el curso de mecanografía de Input está diseñado para ir aumentando metódicamente tu destreza, de forma que vayas obteniendo resultados en cada etapa a medida que vas progresando. Pero no se trata de ningún proceso mágico y requiere un tiempo preciso.

Por ello hemos fraccionado el curso en una serie de etapas fáciles, añadiendo una parte de programa en cada una de las etapas. De esta forma, cuando estés listo para avanzar un paso más, puedes incorporar la parte de programa correspondiente al paso al siguiente nivel de eficiencia. Y a medida que vayas avanzando, irás actualizando constantemente tu velocidad y tu precisión.

APRENDIZAJE DE LAS TECLAS DE REFERENCIA

La habilidad en mecanografía consiste en pulsar en cada momento la te-

cla adecuada sin tener que estar constantemente mirando al teclado. Es más fácil de lo que parece, pero sólo se puede llegar a ello empezando de la forma correcta y conservando los buenos hábitos una vez que hayas llegado a adquirirlos.

Evidentemente, si tienes que llegar a la tecla correcta sólo por el tacto, tendrás que aprender dónde está cada tecla. Para ello lo más importante es el teclado, ya que en caso contrario no tendrías forma de saber dónde tienes cada dedo para presionar o no las distintas teclas. Los mecanógrafos profesionales consiguen esto, asegurándose de que sus dedos siempre están colocados sobre las mismas teclas, llamadas las teclas de referencia (a la teclas A y Ñ se les llama «teclas guía»). Son las teclas de la fila central, que empieza por ASDF... Por eso son éstas las primeras teclas que aprenderás. Una vez que domines las teclas de referencia, puedes llegar a las restantes partes del teclado moviendo tus dedos sólo una corta distancia cada vez.

Para obtener el máximo provecho del curso, tienes que intentar atenerte a él y no hacer trampas. Pero incluso si eres un mecanógrafo de «dos dedos» podrás ver cómo puede mejorar tu habilidad.

Cuando hayas tecleado el programa, tarea que te irá resultando más fácil a medida que vayas avanzando en el curso, y lo ejecutes, siéntate bien, con las manos correctamente colocadas sobre el teclado, listo para empezar, y sigue las indicaciones de pantalla que explicamos más adelante. El dedo meñique de tu mano izquierda debe estar situado sobre la tecla A, el siguiente dedo sobre la S, y así sucesivamente, con el dedo índice colocado sobre la F. El dedo índice de tu mano derecha debe estar sobre la tecla J, el dedo medio sobre la K, etcétera. Esta disposición deja dos teclas sin cubrir en la parte central: la G y la H. Puedes llegar a la G con tu dedo índice izquierdo y a la H con tu dedo índice derecho.

Si todo esto te parece complicado, no te preocupes. La primera parte del programa está diseñada para que te familiarices más con ellos.

FUNCIONAMIENTO DEL PROGRAMA

Al ejecutar el programa, la pantalla te pedirá que selecciones una opción entre las cinco disponibles. Debes empezar por la lección 1 y dominarla antes de pasar a la lección 2, y lo mismo para las restantes lecciones.

Nivel 1

La pantalla presenta las teclas de referencia en una línea, tal como aparecen en el teclado. Aparecerá un asterisco por turno encima de cada tecla, empezando por la A y terminando por la L. Teclea secuencialmente cada tecla cuando el asterisco se va poniendo sobre ella, utilizando el dedo que le corresponde: recuerda que para teclear la G y la H únicamente debes mover los dedos índices, y utiliza el meñique derecho para ; o :.

Si cometes una falta, el asterisco se quedará esperándote. Cuando la pulsación de tecla sea correcta, oirás un pequeño pitido, que se convertirá en un molesto zumbido cuando te comas alguna letra o teclees la que no es.

Al final del ejercicio, aparecerá en pantalla el tiempo que has tardado y el número de faltas que has cometido.

Nivel 2

Cuando ya estés familiarizado con la posición de las teclas, puedes pasar a este nivel que es parecido al anterior: la única diferencia es que el asterisco se mueve ahora aleatoriamente de una tecla a otra. No intentes hacer trampas mirando al teclado. También aquí te aparecerá cuando termines el tiempo que has tardado y el número de errores cometidos (en cada realización del ejercicio te aparecerán 20 letras aleatorias).

Nivel 3

Este ejercicio es similar al del nivel 2, dándote las letras al azar de una en una. Pero en este caso la pantalla no te muestra su posición sobre el teclado, sino que las presenta aisladas en el centro de la pantalla. También aquí aparecerán 20 letras en cada realización del ejercicio.

Nivel 4

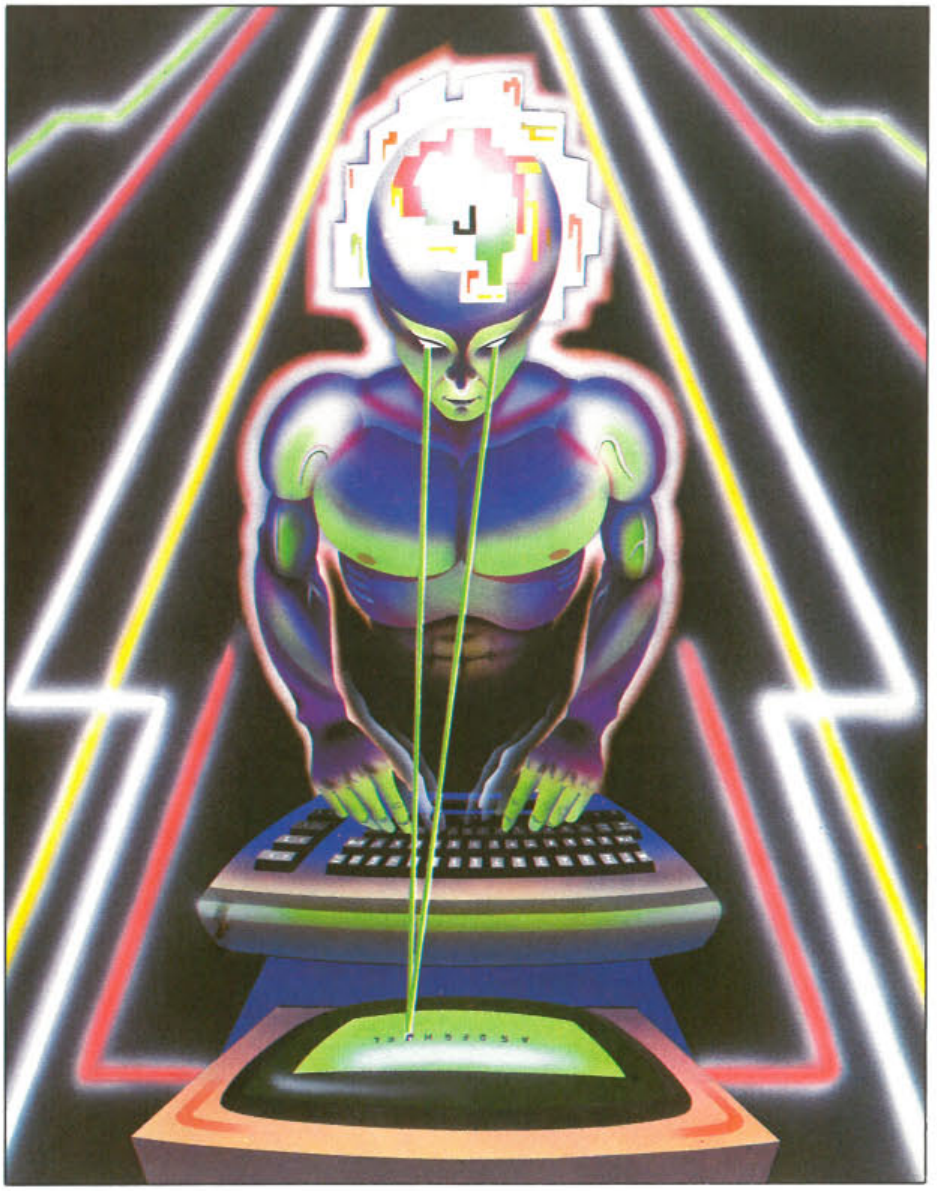
¡Por fin vas a poder teclear algunas palabras! Todas las palabras de este ejercicio están formadas por las letras de las teclas de referencia, por ejemplo SAL. Al igual que en el nivel anterior, aparecerán 20 palabras de forma aleatoria en el centro de la pantalla, una tras otra. A medida que vayas copiando cada palabra, un asterisco irá indicando cada letra. También aquí recibirás al final una puntuación por tu velocidad y por tu precisión. Procura no hacer trampas mirando al teclado o te encontrarás con un hándicap ante algunas de las pruebas más duras que veremos posteriormente en el curso.

Nivel 5

En esta ocasión te aparecerá en pantalla una «frase» de 6 o 7 palabras elegidas al azar. A medida que vayas tecleando las letras, irán apareciendo una tras otra debajo de las palabras impresas. Cuando hayas tecleado la línea completa, se te dirá tu puntuación y la velocidad equivalente en palabras por minuto. Este valor está basado en una longitud media de palabra de 5 letras y un espacio entre cada dos palabras. Cuando teclees las palabras que van apareciendo en pantalla, no te olvides de los espacios.

Practica cada uno de los niveles anteriores hasta que puedas hacer los correspondientes ejercicios con facilidad. Más adelante añadiremos las teclas que faltan y algunas palabras más complicadas.

```
1Ø DIM W$(21),WO$(33):FOR
   Z=1 TO 33: READ
   WO$(Z):NEXT
2Ø FOR Z=1 TO 4Ø:LI$=LI$+"-
   ": NEXT: LI$="[CTRL
   +2]"+LI$
3Ø PRINT "[SHIT+CLR/
   HOME][CTRL+8]"TAB(7)"*****
   MECANOGRAFIA *****"
```



```
4Ø A$="ASDFGHJKL": POKE
   54296,15: GOTO 38Ø
5Ø TI$="ØØØØØØ": S=Ø: WW=1
6Ø N=Ø: POKE 198,Ø
7Ø IF K=5 AND P>Ø THEN
   N=N+1: P=P+1: GOTO 14Ø
8Ø PRINT"[CLR/HOME][8*CRSR
   ABAJO]":IFK<3THENPRINT
   TAB(11)"[CTRL+4][CTRL+
   9]ASDFGHJKL:[CTRL+Ø]'':
   POKE198,Ø
9Ø X=INT (RND(1)*1Ø)+1:
   N=N+1: P=P+1: IF K=1
   THEN X=N:GOTO 12Ø
1ØØ IF K=3 THEN X=INT
   (RND(1)*1Ø)+1: PRINT TAB
   (18)"[CTRL+4][SHIFT+U]
   [SHIFT+C][SHIFT+I][CRS
   RABAJO][3*CRSRIZQDA.][
   SHIFT+B][CTRL+6]"MID$
   [A$,X,1]"[CTRL+4][SHIF
   T+B][CRSRABAJO][3*CRS
   RIZQDA.][SHIFT+J][SHIFT
   +C][SHIFT+K]"
11Ø IFK=4THENPRINTTAB(16
   )"[CTRL+4][7ESPACIOS][7
   *CRSRIZQDA.]"W$(WW):X
   =N*.5+3
12Ø IF K<3 OR K=4 THEN
   PRINT "[CLR/
   HOME][7*CRSR ABAJO]" TA
   B(11+(X-1)*2)"[CTRL+2]*"
13Ø IF K=5 THEN FOR Z=1 TO
   5: PRINT "[CTRL+8]"W$(Z):
   : NEXT Z: PRINT: PRINT
   "[CTRL+9][CTRL+4]'';
14Ø GET K$:IF K$="" THEN
   14Ø
15Ø IF K>3 AND
   K$=MID$(W$(WW),N,1)
   THEN 21Ø
16Ø IF K>3 THEN 18Ø
17Ø IF K$=MID$(A$,X,1) THEN
   21Ø
18Ø W=54276: A=54277
19Ø POKE W,33:POKE A,5Ø:
   POKE 54273;3,cz0:S=S+1
2ØØ POKE 54273,Ø: POKE W,
   32: GOTO 14Ø
21Ø W+54276: A=54277
22Ø IF K<3 OR K=4 THEN
   PRINT TAB(11+(X-
   1)*2)"[CRSR ARRIBA]"
23Ø IF K=5 THENPRINT K$;
24Ø POKE W,33:POKE A,5Ø:
   POKE 54273 ,15Ø
25Ø POKE 54273,Ø: POKE W,32
26Ø IF K=1 AND N=1Ø THEN
   29Ø
27Ø IF K>3 AND
   N<LEN(W$(WW)) THEN7Ø
28Ø IF K<4 AND N<2Ø THEN
   7Ø
29Ø IF K>3 THENNEXT WW
3ØØ WW=VAL (MID$(TI$,3,2)) *
   6Ø+VAL(RIGHT$(TI$,2))
31Ø PRINT"[CTRL+Ø][SHIFT+
   CLR/HOME]"LI$;:PRINT"[
   CTRL+8][6ESPACIOS]HAS
   TARDADO:[CTRL+5]"WW"
   SE9."
32Ø PRINT "[CTRL+8]
   ERRORES COMETIDOS:
   [CTRL+5]"S
33Ø IF K>3 THENPRINT "[CRSR
   ABAJO]"TAB(7) INT (1Ø*NU
   /WW)"[CTRL+2]PALABRAS
   PORMINUTO."
34Ø PRINT LI$: GOTO 53Ø
35Ø NU=Ø: P=Ø: S=Ø FOR
   WW=1 TO
   MM:W$(WW)=WO$(INT
   (RND(1)*33)+1)
36Ø IF K=5 AND WW<>7 THEN
   W$(WW)=W$(WW)K+'' ''
37Ø NU=NU+LEN(W$(WW)):
```

```
NEXT WW:TI$="ØØØØØØ": FOR
WW=1 TO MM: GOTO 6Ø
38Ø POKE 5328Ø,6: POKE
     53281,Ø: POKE 198,Ø
39Ø PRINT "[CLR/
     HOME][1Ø*CRSR
     ABAJO][CTRL+2]"
     TAB(15)"OPCIONES [CRSR
     ABAJO][CTRL+6]"
4ØØ FORZ=1TO5:PRINTTAB(1
     3)Z;":TEST";Z:NEXTZ
41Ø PRINT
     TAB(12)"[CTRL+2][CRSR
     ABAJO]ENTRAR OPCION..."
42Ø GET K$:K=VAL(K$):IFK<1
     OR K> THEN42Ø
43Ø PRINT "[SHIFT+CLR/
     HOME]"LI$"[CTRL+6]"
44Ø IF K<3 THENPRINT "[CRSR
     ARRIBA]PULSA LA TECLA
     INDICADA CON EL
     ASTERISCO";
45Ø IF K=3 THENPRINT "[CRSR
     ARRIBA]TECLEA LA LETRA
     QUE APARECE EN
     PANTALLA.";
46Ø IF K=4 THENPRINT "[CRSR
     ARRIBA]TECLEA LA
     PALABRA QUE SALE EN
     PANTALLA[2 ESPACIOS]";
     :MM=2Ø
47Ø IF K=5 THENPRINT
     TAB(12)"[CRSR
     ARRIBA]TECLEA ESTAS
     PALABRAS":MM=7
48Ø PRINT LI$:FOR Z=1 TO
     1ØØØ: NEXT
49Ø PRINTTAB(11)"[CTRL+2]P
     ULSAUNATECLAPARAEMP
     EZAR":POKE198,Ø:WAIT1
     98,1
5ØØ PRINT TAB(11)
     "[CRSR ARRIBA]
     [29 ESPACIOS]"
51Ø ON KGOTO 5Ø,5Ø,5Ø,35Ø,
     35Ø
52Ø GOTO 38Ø
53Ø FOR Z=1 TO 1ØØØ: NEXT Z:
     GOTO 38Ø
54Ø DATA AL,ALA,ASA,SALA,
     SALAS,SAJAS,AJADA,
     FAJADA,FAJA,ADA,LA,FA,
     GAS,GAFA
55Ø DATA GAFAS,JAFA,JALA,
     SAGA,ALADAS,FLASH,AS,
     LADA,GALA,GALAS,HADA,
     DAS,DAGA
56Ø DATA GASA,LASA,GAL,
     ALAGA,LLAGA,FALSA
```

Cuando ya hayas adquirido un dominio razonable de la fila central del teclado, por ejemplo, conseguir escribir unas 15 palabras por minuto sin cometer ni una sola falta, puedes pasar a la siguiente fase.

LAS TECLAS QWERTY

Si añades las siguientes líneas al programa anterior, puedes enriquecer tu repertorio con las teclas de la línea superior, la línea «QWERTY» de las que toma su nombre este tipo de teclado. Carga el programa con LOAD tal como está y teclea a continuación estas nuevas líneas. Algunas de ellas quedarán escritas por encima de otras que ya no hacen falta, otras son completamente nuevas.

```
1Ø DIMW$(21),WO$(28):FORZ
   =1TO28:READWO$(Z):NEXT
4Ø A$="QAWSEDRFTGGHUJIK
   OLP:'':POKE54296,15:
   GOTO38Ø
```

```
8Ø PRINT "[CLR/HOME][8*CRSR
   ABAJO]":IF K<3 THENPRINT
   TAB(11)"[CTRL+4][CTRL+9
   ]"A$:POKE198,Ø
9Ø X=INT (RND(1) * 2Ø) +
   1:N=N + 1:P=P + 1:IF K=1
   THENX=N:GOTO 12Ø
1ØØ IF K=3 THENX=INT
   (RND(1) * 2Ø/) + 1:PRINT T
   AB(18)"[CTRL+4][SHIFT+
   U][SHIFT+C][SHIFT+I][CR
   SRABAJO][3*CRSRIZQDA.]
   [SHIFT+B][CTRL+6]"MI
   D$(A$,X,1)"[CTRL+4][SHI
   FT+B][CRSRABAJO][3*CR
   SRIZQDA.][SHIFT+J][SHIF
   T+C][SHIFT+K]"
11Ø IFK=4THENPRINT
   TAB(16)"[CTRL+4][12
   ESPACIOS][12*CRSR
   IZQDA.]"W$(WW):X=N+5
12Ø IFK<3ORK=4THENPRINT
   "[CLR/HOME][7*CRSR
   ABAJO]"TAB(11+(X-
   1))"[CTRL+2]*"
21Ø W=54276:A=54277
22Ø IFK<3ORK=4THENPRINT
   "[CLR/HOME][7*CRSR
   ABAJO]"TAB(11+(X-1))""
26Ø IFK=1ANDN=2ØTHEN29Ø
35Ø NU=Ø:P=Ø:S=Ø:FORWW=
   1TOMM:W$(WW)=WO$(IN
   T(RND(1)*28)+1)
36Ø IFK=5ANDWW<>5THENW
   $(WW)=W$(WW)+"
47Ø IFK=5THENPRINT
   TAB(12)"[CRSR
   ARRIBA]TECLEAESTAS
   PALABRAS":MM=5
54Ø DATASEDA,FRESA,LOSA,J
   URAS,PIERDO,TIPOS,TOR
   ERO,PITIDO,RELIQUIA,HA
   LITO
55Ø DATALITROS,POLIPOS,TRI
   LITA,FERIA,GRATIS,PEPIT
   A,LUISILLO,TAURO,LEO
56Ø DATAQUERIDOS,ARIES,SA
   GITARIO,TERESA,GERARD
   O,DESEOS,TARJETA,GAST
   OS,LOGROS
```

Al ejecutar el programa, te aparecerá en pantalla el menú de cinco opciones que ya conoces. El nivel 1 presenta la secuencia de letras QAW SEDRFTGYHUJIKOLP: Observa que también se incluye al final el carácter «:». Como invitación a teclear, te aparecerá el asterisco por encima de dichas teclas, cambiando de izquierda a derecha igual que antes.

Los niveles 2 y 3 funcionan exactamente de la misma manera que en la primera prueba, haciendo que teclees estos caracteres de forma aleatoria pero con un conjunto de letras más amplio.

Los niveles 4 y 5 son más difíciles que antes, ya que en ellos se te pide que teclees palabras más largas y más difíciles, cambiando de una filas de teclas a otra dentro de cada palabra.

Siéntate bien, con tus dedos correctamente colocados sobre las teclas de referencia e intenta pulsar las teclas de la fila de arriba, moviendo únicamente el dedo necesario para pulsar cada tecla y sin mover el resto de la mano. Esta vez tendrás que utilizar el dedo meñique de tu mano izquierda para la Q, además de para la A, el siguiente dedo para la S y la W, etcétera, hasta llegar al dedo meñique de tu mano derecha que utilizarás para ; y para P. También en este caso tus dedos índices tendrán más trabajo que los demás; tienes que utilizar el índice de tu mano izquierda para F, G, R y T mientras que el índice derecho se encargará de pulsar H, J, Y y U. En el momento en que hayas pulsado una tecla perteneciente a la fila de arriba, el dedo correspondiente debe volver a su posición sobre su tecla de referencia.

LAS TECLAS ZXCV

Cuando hayas adquirido una cierta práctica en el uso de las teclas de la fila superior mezcladas con la fila central, habrá llegado el momento de pasar a la última fila.

La parte tercera del curso te enseña la manera de usar las líneas de fondo y central combinadas, progresando a través de los cinco niveles de habilidad, pero también sin mezclar las tres filas de teclas.

Aquí tienes las modificaciones que debes introducir en el programa.

```
4Ø A$="AZSXDCFVGBHNJMK,
   L.:/"POKE54296,15:GOTO
   38Ø
54Ø DATA CASA,CASACA,
   CASADA,MAZA,MALAGA,
   MANA,JAMAS,VACAS,BALA,
   ANDAS,CANGAS
55Ø DATA BLANDAS,BLAS,
   FACHA,HADAS,ALBAS,
   CLAMAN,CLAVAN,ZAFAS,
   AVAL,MALDAD
56Ø DATA SACAN,NADAS,
   DAMAS,BANANA,LLAMAS,
   FLAN,CAMAS
```

Al ejecutar ahora el programa, podrás ir progresando a lo largo de las cinco lecciones, pero con letras y palabras sacadas de las dos filas inferiores del teclado.

Una vez más, siéntate correctamente y coloca los dedos sobre las teclas de referencia. Esta vez deberás desplazar tus dedos hacia la fila de abajo del teclado volviendo a colocarlos inmediatamente sobre su tecla de referencia. El dedo meñique de tu mano izquierda actuará sobre la A y la Z, el anular sobre la S y la X, etcétera. El dedo índice de la mano izquierda se utilizará para las letras F, G, V y B mientras que el índice derecho se encargará para H, J, N y M. Los otros dedos tienen que actuar sobre las teclas de puntuación de la línea de fondo, que suelen variar de unos teclados a otros.

EL ALFABETO COMPLETO

Cuando tu habilidad en la utilización de las teclas de la fila inferior, sea comparable a la que tienes para la fila superior, puedes pasar a la siguiente etapa del curso. Esta vez puedes practicar utilizando todo el alfabeto.

Aquí tienes las alteraciones que tienes que hacer en tu programa anterior:

```
4Ø A$="QAZWSXEDCRFVTGBY
   HNUJMIK,O,L.P:/":POKE542
   96,15:GOTO38Ø
8Ø PRINT "[CLR/HOME][
   8*CRSR ABAJO]":IF K<3
   THENPRINT TAB(5)"[CTRL+
   4][CTRL+9]"A$:POKE198,Ø
9Ø X=INT (RND(1) * 3Ø) +
   1:N=N + 1:P=P+1 : IF K=1
   THENX=N:GOTO 12Ø
1ØØ IF K=3 THENX=INT
   (RND(1) * 3Ø) + 1:PRINT
   TAB(18)"[CTRL+4]
   [SHIFT+U][SHIFT+C]
   [SHIFT+I][CRSR ABAJO]
   [3*CRSR IZQDA.][SHIFT+B]
   [CTRL=6]"MID$(A$,X,1)"
   [CTRL=4][SHIFT=8][CRS
   R ABAJO][3*CRSR IZQDA.]
   [SHIFT+B][CTRL+6]"MID$
   (A$,x,1)"[CTRL+4][SHIFT
   +B][CRSR ABAJO][3*
   CRSR IZQDA.][SHIFT+J][
   SHIFT+C][SHIFT+K]"
11Ø IFK=4THENPRINT
   TAB(16)"[CTRL+4][12
   ESPACIOS][12*CRSR
   IZQDA.]"W$(WW):X=N+11
12Ø IFK<3ORK=4THENPRINT
   "[CLR/HOME][7*CRSR
```

```
      ABAJO]"TAB(5+(X-
      1))"[CTRL][CTRL+2]*"
21Ø   W=54276:A=54277
22Ø   IFK<3ORK=4THENPRINT
      "[CLR/HOME][7*CRSR
      ABAJO]"TAB(5+(X-1))""
26Ø   IFK=1ANDN>3ØTHEN7Ø
28Ø   IF(K<4ANDK>1)ANDN<2Ø
      THEN7Ø
54Ø   DATACESAR,BUSCAN,
      LOGRAN,LLAVES,TOMATES,
      LIBROS,BIGOTES,FALDAS,
      BLUSAS
55Ø   DATAPINTURA,COMIDA,
      LLAVERO,COLILLA,ROJO,
      AMARILLO,VERDE,AZUL,
      VERDE
56Ø   DATACOSAS,GATAS,AJILLO,
      IRAK,COLOR,TIGRE,GATO,
      PERRO,LISTO,DADOS
```

Como ocurre con todos los demás programas de este curso de mecanografía, cuando te familiarices con las palabras que figuran en las sentencias DATA, conviene que las cambies por otras nuevas. Sobre todo ten cuidado de no utilizar menos palabras que en el programa original, para no obtener un mensaje de error del tipo de OUT OF DATA. Naturalmente, puedes tener más entradas, pero no serán leídas por el ordenador a menos que cambies el programa.

Recorre progresivamente las cinco lecciones, igual que antes. Acuérdate de que tus dedos regresen a su posición correcta sobre las teclas de referencia después de pulsar una tecla en la fila de arriba o en la de abajo.

MEJOR TODAVIA

A medida que el curso va progresando, irás teniendo la oportunidad de mejorar tu precisión y tu velocidad sobre todas las teclas de letras del teclado. Veremos también la manera de incluir las teclas numéricas y los signos de puntuación, lo cual resulta crucial para la introducción correcta de los listados de programas.

Añadiendo todas estas teclas y las teclas de control, tal como SHIFT, te encontrarás en seguida en condiciones de teclear con gran comodidad frases complejas y líneas de programa largas; veremos además una gran cantidad de ejercicios que te permitirán hacer prácticas para aumentar aún más tu destreza.

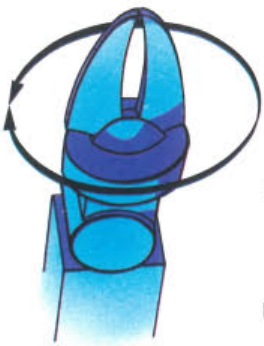
P y R

¿Qué velocidad debo intentar alcanzar para empezar?

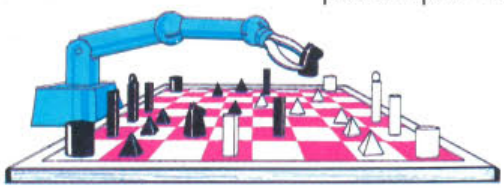
En esta primera fase del curso, tienes que preocuparte más de la exactitud que de la velocidad. Cuando ya consigas teclear todas las letras y todas las palabras sin errores, puedes intentar adquirir algo más de velocidad.

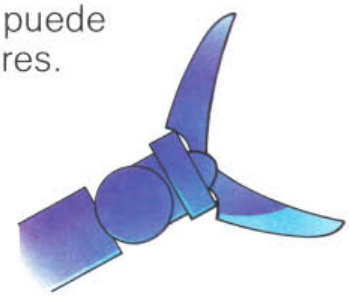
Tienes que practicar con los niveles 1, 2 y 3 hasta que estés en condiciones de teclear las 20 letras en unos 12 o 13 segundos; para los niveles 4 y 5 debes intentar escribir unas 15 palabras o más por minuto.

!PARTICIPA EN EL CONCURSO!



En INPUT estamos convencidos de que aún puedes hacer muchas más cosas con tu ordenador. Sin duda, muchos lectores estareis utilizando vuestro micro para funciones de lo más variadas, en unos casos; pintorescas, en otros; mientras que algunos listillos habrán podido utilizarlo para resolver tareas complejas. Es lógico, modificando programas y variando los periféricos nuestro ordenador puede prestar sus servicios en infinidad de facetas. INPUT quiere que esas aplicaciones y utilidades a las que has conseguido dedicar tu ordenador, sean conocidas por todos sus lectores y por eso ha organizado el «Concurso de Aplicaciones y Utilidades», en el que puede participar cualquiera de nuestros lectores.





BASES

UTILIDADES Y APLICACIONES: Si tu ordenador controla la calefacción de tu casa, gobierna un robot, dirige un pequeño negocio, organiza la maqueta de tu tren eléctrico, o cualquier cosa interesante u original; envíanos información gráfica y listados de tus programas, grabados en un cassette, diskette o microdrive.

Todo ello habrá de venir acompañado por un texto que aclare cuál es su objetivo, el modo de funcionamiento y una explicación del cometido que cumplen las distintas rutinas que lo componen. El texto se presentará en papel de tamaño folio y mecanografiado a dos espacios. No importa que la redacción no sea muy clara y cuidada; nuestro equipo de expertos se encargará de proporcionarle la forma más atractiva posible.

UN JURADO propio decidirá en cada momento qué colaboraciones reúnen los requisitos adecuados para su publicación, y evaluará la cuantía del premio en metálico al que se hagan acreedoras.

No olvideis indicar claramente para qué ordenador está preparado el material, así como vuestro nombre y dirección y, cuando sea posible, un teléfono de contacto. Entre todos los trabajos recibidos durante los próximos tres meses SORTEAREMOS:

- **Un premio de 50.000 ptas.**
- **Un premio de 25.000 ptas.**
- **Un premio de 10.000 ptas. en material microinformático a elegir por los afortunados.**

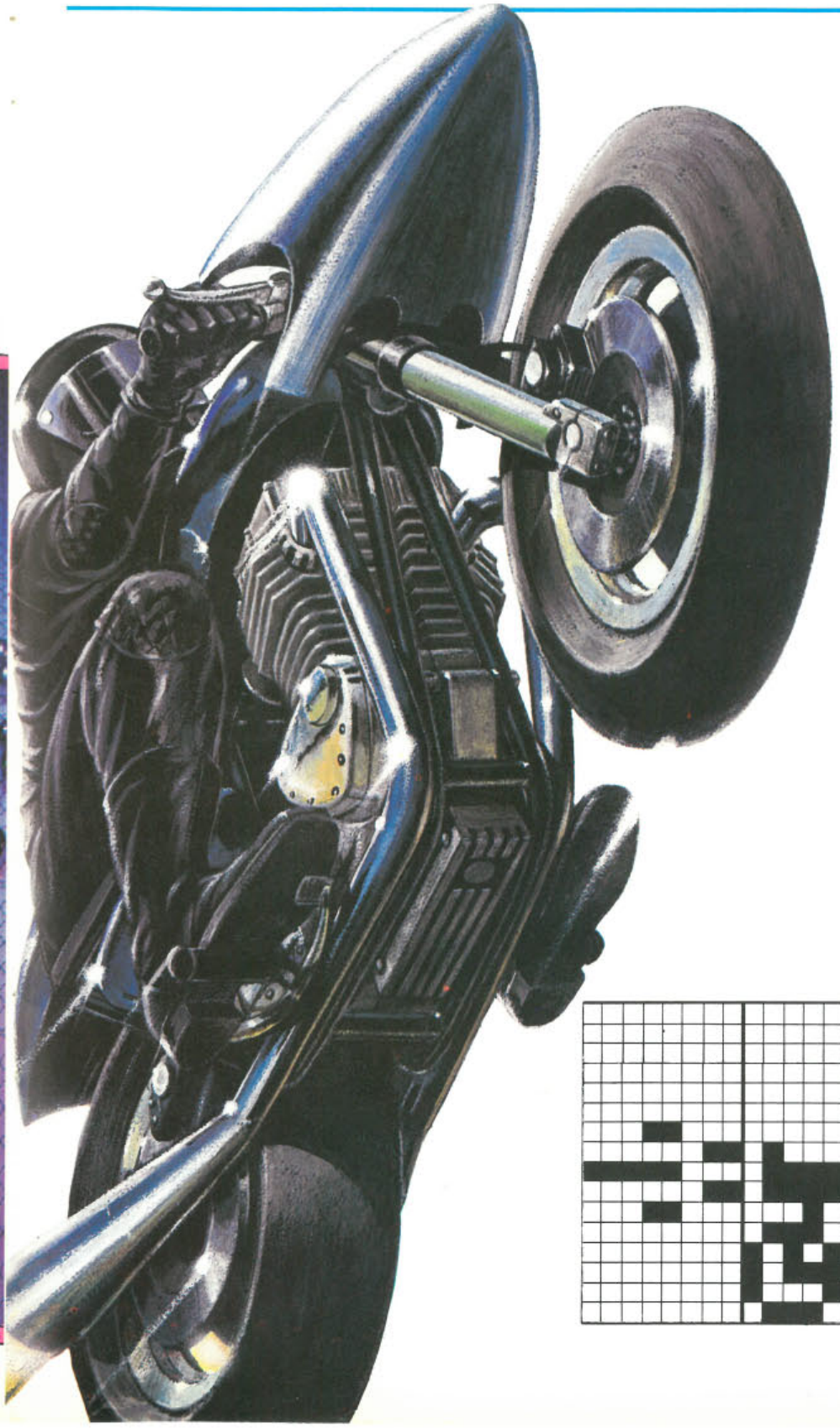
¡No os desanimeis!, por muy simples o complejas que puedan parecer vuestras ideas, todas están revisadas con el máximo interés.

INPUT COMMODORE
Aribau, 185. Planta 1.ª
08021 BARCELONA

NOTA: INPUT no se responsabiliza de la devolución del material que no vaya acompañado por un sobre adecuado con el franqueo correspondiente.

MOVIMIENTO DE IMAGENES EN EL VIC-20

- CREACION DE UNA MOTO Y UN SUBMARINO
- MOVIMIENTO DE LOS CARACTERES



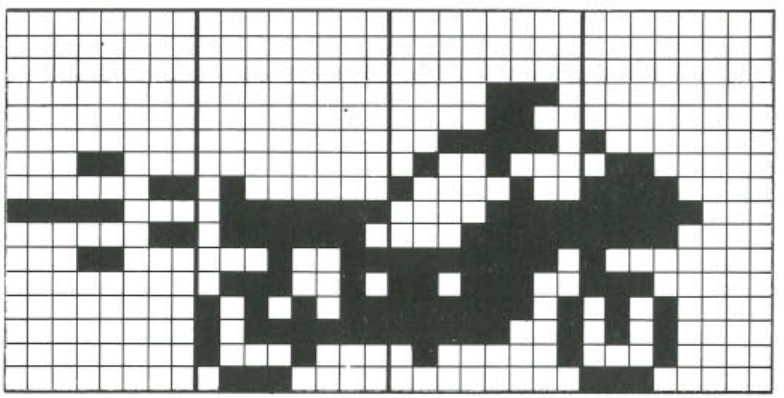
Aquí presentamos a los poseedores de un Vic-20 la posibilidad de crear gráficos animados sencillos. Puedes utilizar las rutinas tal como están o seguir las instrucciones para crear tus propios caracteres.

Ya hemos visto anteriormente unos gráficos sencillos en lenguaje máquina para el Commodore 64. Ahora le toca el turno a los propietarios de un Vic.

Realmente este ordenador no posee las sofisticadas habilidades gráficas del Commodore 64 y de hecho no se puede considerar como un ordenador para gráficos. Pero como verás, hay formas de utilizar el código máquina para generar algunos efectos de pantalla sencillos pero impresionantes.

El Vic-20 no utiliza los *sprites* gráficos como el Commodore 64, ni te obliga a definir una retícula antes de empezar a construir los GDU's. Los gráficos del Vic se generan redefiniendo el conjunto de caracteres.

Cada número, letra del alfabeto o símbolo que figura en el teclado, aparece sobre la pantalla como un conjunto de pixels. Se suelen dimensionar normalmente dentro de cuadrados de 8×8. Los pixels que están activados, apareciendo en claro sobre la pantalla, constituyen el conjunto que da forma al símbolo. Los restantes pixels, desactivados u oscuros, constituyen el fondo de pantalla.



Para construir algún diseño propio, no tienes más que redefinir qué pixels tienen que activarse en una determinada letra para formar parte del dibujo y a continuación construir toda la figura utilizando la sentencia PRINT con todos los caracteres redefinidos. El número de caracteres necesarios depende de la complejidad de la imagen que quieras conseguir y del detalle que puedas obtener dentro de cada carácter.

Al construir una figura como la del motorista acróbata que aparece en la página de al lado, podrías dividir toda la figura en ocho pequeños cuadrados de 8×8 y dar forma a todo el dibujo de esta manera. Pero con el Vic-20 puedes conseguir caracteres de doble altura. Utilizando esta posibilidad, sólo tienes que redefinir cuatro caracteres de 8×16 para cada figura.

Cada carácter de tamaño natural se define por medio de ocho números, uno por cada fila de pixels. Estos números se calculan exactamente de la misma manera que los números que definen los GDU's en otros micros domésticos. Pero hacen falta 16 números para definir un carácter de doble altura, uno para cada fila. También en este caso se calculan de la misma forma.

Los números se leen en una zona protegida de la memoria del Vic y se almacenan como código máquina para su posterior manipulación por las líneas del BASIC al hacer que la figura se mueva.

El siguiente programa crea y mueve el motorista acróbata por la pantalla:

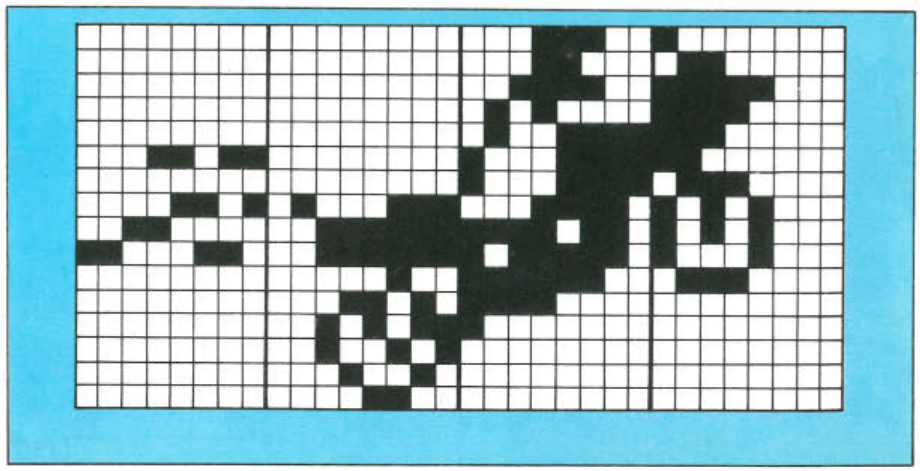
```
1Ø DATA Ø,Ø,Ø,Ø,Ø,Ø,24,3,248,
   24,Ø,Ø,Ø,Ø,Ø,Ø
12 DATA Ø,Ø,Ø,Ø,Ø,Ø,64,127,
   127,19,114,171,191,136,
   112
14 DATA Ø,Ø,Ø,14,12,62,72,
   132,15,31,254,22Ø,253,
   249,65,Ø
16 DATA Ø,Ø,Ø,Ø,Ø,128,96,24Ø,
   248,24Ø,128,224,8Ø,8Ø,16,
   224
18 DATA Ø,Ø,Ø,Ø,Ø,27,Ø,13,48,
   198,Ø,Ø,Ø,Ø,Ø,Ø
2Ø DATA Ø,Ø,Ø,Ø,Ø,Ø,Ø,71,63,
   63,4,26,43,37,18,12
22 DATA 28,24,6Ø,8Ø,79,143,
   143,3Ø,246,19Ø,252,24Ø,
   128,Ø,Ø,Ø
24 DATA 128,96,248,24Ø,224,
   192,112,168,168,136,112,
   Ø,Ø,Ø,Ø,Ø
1ØØ FOR Z=Ø TO 15+7*16:
   READX: POKE 6144+Z;
   X:NEXT
1Ø5 FOR Z=Ø TO 15: POKE
   6656+Z,Ø: NEXT
11Ø POKE 36867,255: POKE
   36869,254:
   A$="[CTRL+7]ABC":
   B$="[CTRL+7]EFG":
   C$="@D"
12Ø POKE 36879,25: POKE
   36878,15
125 PRINT "[SHIFT + CLR/
   HOME]"
2ØØ FOR Z=Ø TO 255: GET Z$
21Ø PRINT "[CLR/HOME]"
   TAB(Z)" ";: POKE 646;
   RND(1)*3+2: PRINT
   MID$(C$,RND(1)*2+1,1);
22Ø IF Z$="" THEN PRINT A$:
   POKE 36877,
   2ØØ+RND(1)*5: S=S+2Ø
23Ø IF Z$="[CURSOR
   IZQUIERDA]" AND Z<23Ø
   THEN PRINT "[SHIFT +
   CLR/HOME]";: Z=Z+22:
   PRINT TAB(Z)"[CTRL +
   3]@" A$: GOTO 26Ø
24Ø IF Z$="[CURSOR
   DERECHA]" AND Z>3Ø
   THEN PRINT "[SHIFT +
   CLR/HOME]";: Z=Z-22:
   PRINT TAB(Z)" [CTRL + 3]"
   B$: GOTO 26Ø
25Ø IF Z$<>"" THEN PRINT
   B$: POKE 36877,
   24Ø+RND(1)*5: IF S>Ø
   THEN S=S-4Ø
26Ø FOR C=1 TO S: NEXT C,Z:
   GOTO 125
```

Las líneas 10 a 24 contienen los datos que definen la forma de la imagen. Cada línea define un carácter diferente, de manera que hay 16 elementos de datos en cada una. Si quieres modificar la imagen, puedes alterar estos datos para redefinir cada carácter y en consecuencia lo que está siendo representado en pantalla.

La línea 1ØØ lee los datos y los pokea en las direcciones de memoria 6144 a 6271. La línea 1Ø5 llena el espacio de caracteres con ceros para garantizar que tengas una rutina de blanqueado en la que no se imprima nada en el fondo de la pantalla.

En la línea 11Ø, al pokear 36867 con el valor 255 se aprovecha la posibilidad del carácter de doble altura, y al pokear 36869 con 254, se redirecciona de nuevo el puntero de caracteres de memoria para asegurar que se está utilizando el conjunto particular de caracteres que tú has redefinido. Hay otros dos conjuntos de caracteres que comienzan en 512Ø y 7168 a los que puedes acceder pokeando 253 y 255 respectivamente en 36869.

A$ y B$ ponen juntos los caracteres ABC y EFG para formar las dos fi-



guras del motorista, una conduciendo normalmente y la otra haciendo el caballito como se muestra en la figura. La flecha inversa que figura al frente de estas cadenas de caracteres hace que el Vic dibuje la motocicleta de color azul. La cadena C$ contiene los dos caracteres de llamarada.

En la línea 12Ø se pokea 36879 con 25, haciendo que el borde de pantalla sea blanco y al pokear 36878 con el valor 15 se establece un valor para el volumen sonoro. La línea 2ØØ define un bucle FOR...NEXT que sirve para mover la moto por la pantalla, mientras que GET$ deja la máquina lista para esperar las pulsaciones de tecla que controlan el movimiento.

Al imprimir una S inversa al principio de la línea 21Ø el cursor vuelve a la posición de «home», lo que evita que se produzca un *scrolling* de pantalla. Al pokear 646 se cambia el color del siguiente carácter —en este caso de forma aleatoria— y el resto de la línea imprime aleatoriamente uno de los dos caracteres de llamarada.

Las siguientes cinco líneas sirven para presentar y mover la moto. Pulsa la tecla de cursor a derecha izquierda y la moto se desplazará hacia abajo sobre la pantalla, mientras que si pulsas la tecla de cursor hacia arriba y abajo, la moto se encabritará y empezará a moverse hacia arriba por la pantalla. Si pulsas cualquier otra tecla, harás que acelere, avance hacia adelante y haga el caballito.

El signo de £ inversa hace que la llamarada sea roja cuando la moto se mueve hacia arriba o hacia abajo, y de un color aleatorio en los demás casos. El POKE de 36877 hace que se produzcan los efectos sonoros.

EL ZOCO DE INPUT

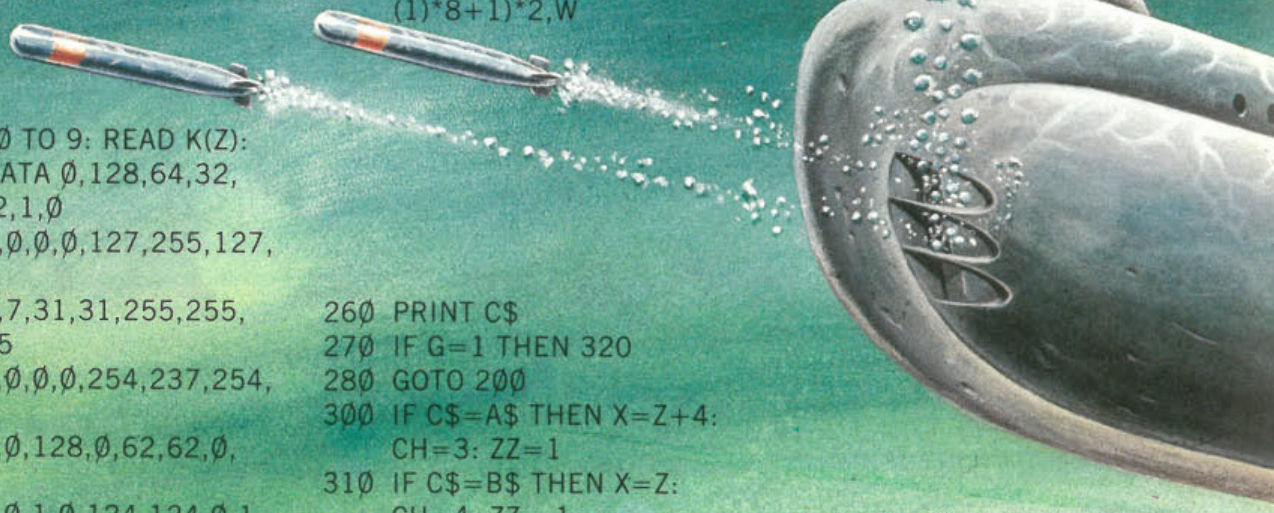
Todo se compra y se vende. Los antiguos zocos fueron lugares destinados a todo tipo de transacciones. INPUT también tiene el suyo. Vuestras operaciones de compra, cambio o venta serán publicadas en esta sección, pero dos son las limitaciones que imponemos:

a) La propuesta tendrá que ver con la microinformática.
b) Nos reservamos el derecho de no publicar aquellos insertos de los que se sospeche un trasfondo lucrativo.

Ahora un ruego. Tratar de resumir al máximo el texto; escribir casi como un telegrama siendo claros y concisos.

EL SUBMARINO

El siguiente programa sólo necesita para construir el submarino la mitad de los datos. El gráfico sólo requiere ocho pixels y no tienes que usar caracteres de doble altura. Una vez más, los datos se introducen a razón de un carácter cada vez, pero ahora sólo necesitas ocho números para definir el carácter. Los diez datos de la línea 5 se utilizan para crear la oleada blanca de puntos aleatorios.



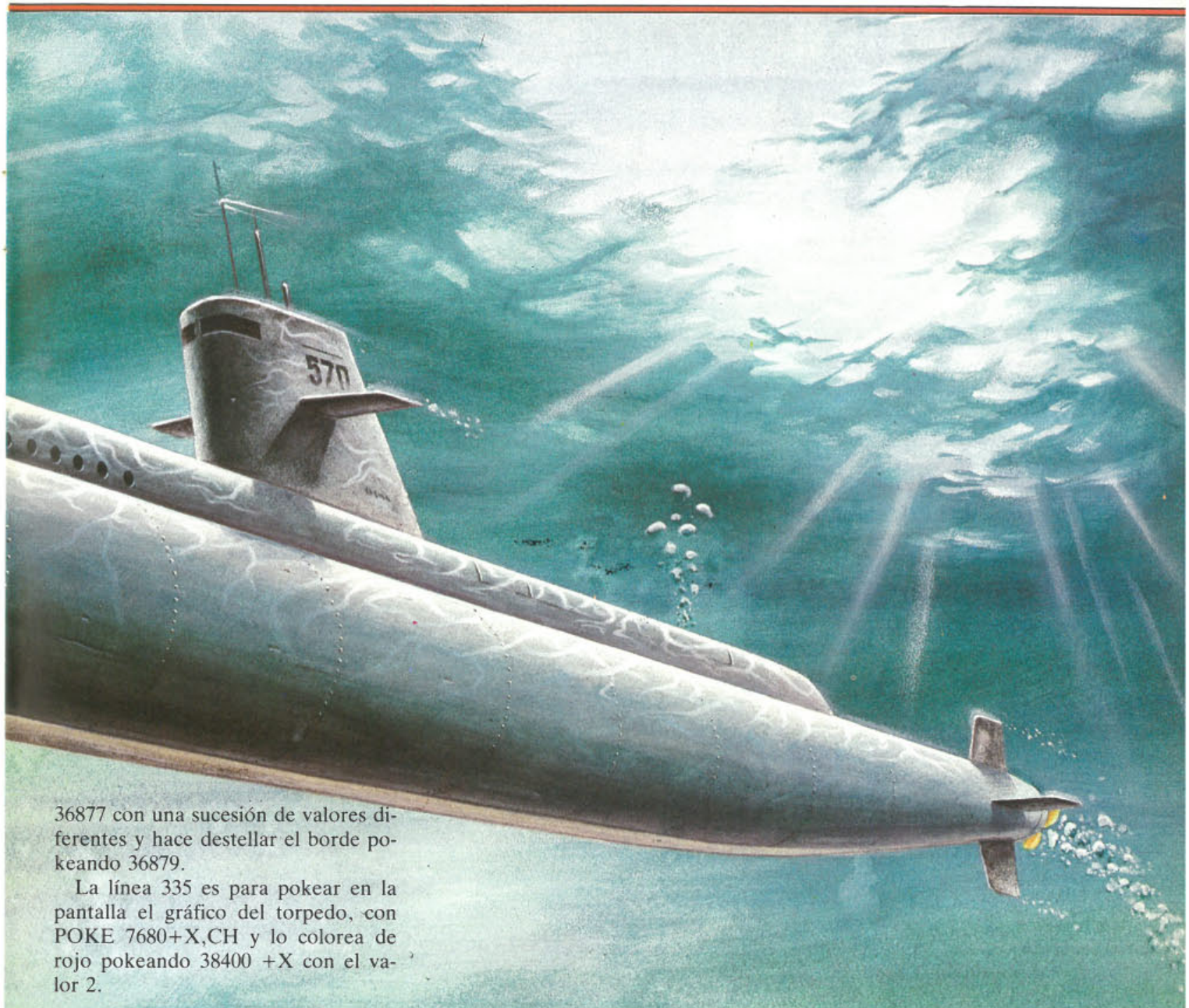
```
 5 FOR Z=Ø TO 9: READ K(Z):
   NEXT: DATA Ø,128,64,32,
   16,8,4,2,1,Ø
1Ø DATA Ø,Ø,Ø,Ø,127,255,127,
63
12 DATA 4,7,31,31,255,255,
   255,255
14 DATA Ø,Ø,Ø,Ø,254,237,254,
   252
16 DATA Ø,Ø,128,Ø,62,62,Ø,
   128
18 DATA Ø,Ø,1,Ø,124,124,Ø,1
2Ø DATA Ø,Ø,Ø,Ø,127,233,127,
   63
22 DATA 32,224,248,248,255,
   255,255,255
24 DATA Ø,Ø,Ø,Ø,254,255,254,
   252
1ØØ FOR Z=Ø TO 7+7*8: READ
    X: POKE 6144+Z,X: NEXT
1Ø5 FOR Z=Ø TO 7: POKE
    64ØØ+Z,Ø: NEXT: POKE
    65Ø,128
11Ø POKE 36867,24: POKE
    36865,6Ø: POKE 36869,
    254: A$"[CTRL + 2] [CTRL
    + 8]@AB [CTRL + 2]":
    B$="[CTRL + 2] [CTRL +
    8]EFG [CTRL + 2]"
12Ø POKE 36879,1Ø9: POKE
    36878,15: C$=A$
125 Z=12Ø: PRINT"@" TAB(Z);
    HOME]" TAB(Z);C$: G=Ø
2ØØ GET Z$
21Ø PRINT "[CLR/HOME]"
    TAB(Z);
22Ø IF Z$="," THEN C$=A$:
    Z=Z+1: IF Z>126 THEN
    Z=126: PRINT
    "[SHIFT+CLR/HOME]"
    TAB(Z);
23Ø IF Z$="." THEN C$=B$:
    Z=Z+1: IF Z<111 THEN
    Z=111: PRINT "<" TAB(Z);
24Ø IF Z$=" " AND G=Ø THEN
    G=1: N=Ø: GOTO 3ØØ
245 W=K(RND(1)*10)
25Ø POKE 64ØØ+INT(RND
    (1)*8+1)*2,W
26Ø PRINT C$
27Ø IF G=1 THEN 32Ø
28Ø GOTO 2ØØ
3ØØ IF C$=A$ THEN X=Z+4:
    CH=3: ZZ=1
31Ø IF C$=B$ THEN X=Z:
    CH=4: ZZ=-1
315 GOTO 335
32Ø N=N+1: POKE 768Ø+X,32:
    POKE 384ØØ+X,1: POKE
    36875,128+N*3:
    X=X=+ZZ
325 IF (X<11Ø OR X>131)
    THEN POKE 36875,Ø: G=Ø
33Ø IFG=Ø THEN FORD= Ø TO
    1Ø9: POKE 36877,D+13Ø:
    POKE 36879,D: POKE
    36877,23Ø-D: NEXT: GOTO
    2ØØ
335 POKE 384ØØ+X,2: POKE
    768Ø+X,CH
34Ø GOTO 2ØØ
```

Este programa se parece bastante al del motorista acrobático. En este caso es la tecla del punto y coma la que hace que el submarino haga un viraje y se mueva en la dirección opuesta. El POKE 650,128 de la línea 1Ø5 activa el efecto de repetición de las pulsaciones de tecla cuando las dejas apretadas. Las líneas 245 y 25Ø proporcionan la distribución aleatoria de los puntos blancos sobre el fondo azul.

La novedad que tenemos aquí es el mecanismo de disparo del torpedo. Cuando se pulsa la barra espaciadora, el ordenador salta a la rutina de fuego. Las líneas 3ØØ y 31Ø miran adónde está apuntando el submarino y deciden cómo hay que lanzar el torpedo, desde dónde tiene que arrancar y qué camino seguirá.

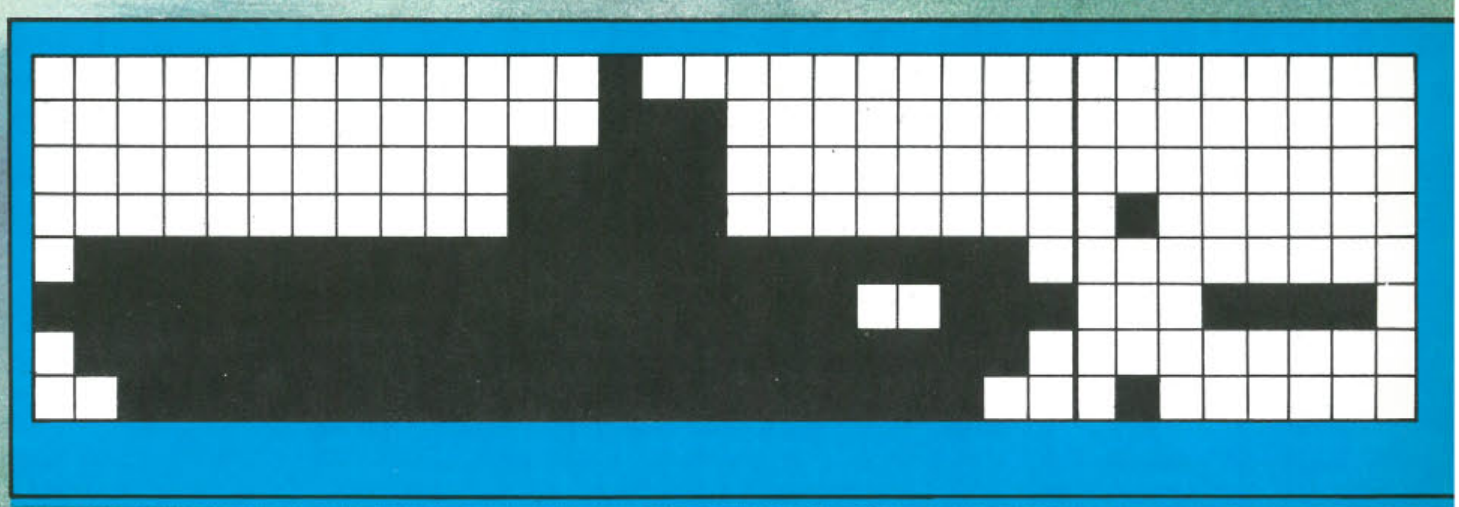
La línea 32Ø mueve el torpedo por la pantalla. El POKE 7680+X,32 sirve para ir borrando lo que queda detrás del torpedo, escribiendo un espacio sobre la vieja imagen. A continuación la instrucción POKE 38400+X,1 pokea en dicho espacio el color del fondo. Por último, POKE 36875 te da el sonido.

La línea 325 sirve para detectar el momento en que el torpedo alcanza el borde de la zona de pantalla, suprime el sonido otra vez y borra el gráfico del torpedo poniendo un Ø en G. Cuando el torpedo choca con el borde, la línea 33Ø da diferentes sonidos pokeando



36877 con una sucesión de valores diferentes y hace destellar el borde pokeando 36879.

La línea 335 es para pokear en la pantalla el gráfico del torpedo, con POKE 7680+X,CH y lo colorea de rojo pokeando 38400 +X con el valor 2.



REFINA TUS GRAFICOS DE PANTALLA

- MAS COMBINACIONES DE COLOR
- IMAGENES CON CIRCULOS Y ARCOS
- EL BASIC DE SIMONS PARA LOS GRAFICOS DE COMMODORE

Los ordenadores domésticos ofrecen muchas posibilidades para el artista en ciernes. Aquí tienes unas cuantas sugerencias para ampliar el uso de los comandos gráficos del BASIC y crear nuevas imágenes en la pantalla.

Una vez que domines los rudimentos del dibujo de imágenes en la pantalla, puedes empezar a ampliar el campo de tus esfuerzos artísticos, utilizando algunos de los comandos gráficos especializados de que dispone tu ordenador. Algunos ordenadores disponen de comandos tales como MOVE, PLOT, DRAW, PAINT y CIRCLE, que permiten dar rienda suelta a la imaginación, creando cualquier tipo de imágenes, desde un gráfico de tipo comercial hasta el escenario de fondo de una emocionante aventura.

Ya hemos analizado anteriormente la manera de utilizar los sencillos comandos de dibujo para crear en la pantalla imágenes a base de líneas y, en determinados casos, añadirle el color. Pero tú puedes ampliar estas técnicas con nuevos métodos para dibujar puntos, líneas, triángulos, cuadrados y círculos. Dichos métodos, utilizados ellos solos o en combinación con los colores, se convierten en un poderoso medio de generación de imágenes estáticas o incluso en movimiento.

La forma en que los ordenadores utilizan estos comandos varía de unos a otros, pero todos son capaces de lograr algunos efectos interesantes. En este sentido, el ordenador Commodore 64 es una excepción, ya que su BASIC estándar no trae estos comandos. Sin embargo puedes disponer de ellos añadiéndole la unidad de BASIC de Simons. Así se ponen a su disposición muchos comandos adicionales que aumentan las posibilidades de la programación. Algunos de estos comandos tienen por objeto simplificar la programación de gráficos y ya nos hemos ocupado anteriormente de la manera de utilizarlos. Dos comandos de dibujo adicionales de los que todavía no nos hemos ocupado son ARC y ANGL. El primero se utiliza para dibujar parte de una circunferencia y toma la forma:

```
99 ARC15Ø,5Ø,6Ø,27Ø,1,
   3Ø,3Ø,1
```

En condiciones normales, este comando deberá ir precedido por el comando HIRES, por lo que también tendrás que teclear la siguiente línea:

```
9Ø HIRESØ,1
```

Los dos primeros números que figuran a continuación de la palabra ARC definen las coordenadas de pantalla del centro del círculo cuyo arco quieres pintar. Estos valores se refieren a las posiciones de los pixels en la configuración estándar, es decir: primero el valor horizontal (X) y a continuación el vertical (Y).

Intenta ajustar una de estas cifras, o las dos, y vuelve a ejecutar el programa varias veces. Observarás que cuanto más alto sea el valor de dichos números, más cerca del fondo o del borde derecho de la pantalla se dibuja la curva, pudiendo llegar a alcanzar dicho borde.

La longitud real del arco está regulada por los dos pares de valores siguientes. El primer par de valores (60 y 270) son los ángulos de comienzo y de parada. Si se tratase de un reloj, los ángulos empezarían a contarse a partir de las 12, con valores que van desde 0 a 360. Así, la curva de nuestro ejemplo empieza a las 2 y termina a las 9 en punto. Prueba a cambiar los valores de este par de números para observar el efecto resultante.

El siguiente número (1) es el valor del incremento; ajustando este valor dentro del intervalo de 1 a 360 puedes elegir valor en grados de cada cambio, que para un valor de 10 resultará algo más tosco que en el programa original. Para hacerte una idea de las posibilidades, introduce las siguientes modificaciones en el programa original:

```
 9Ø HIRESØ,1:FORN=
    1ØTO36ØSTEP1Ø
 99 ARC16Ø,8Ø,Ø,36Ø,N,
    84,6Ø,1
1ØØ NEXT:PAUSE1Ø
```

Aquí puedes ver el efecto resultante de ajustar el incremento en pasos de 10 grados, para una curva que empiece y termine en el mismo punto, es decir, para un círculo. (El uso de la variable N en este ejemplo es un potente método de abordar los cambios, con una cantidad mínima de programación adicional.)

El par de valores que figura detrás de N en este ejemplo, regula la apariencia física del círculo cuyo arco se está dibujando. El primer valor es la longitud, en pixels, del radio X u horizontal; el segundo valor corresponde al radio en la dirección Y o vertical. Si quieres que te resulte un arco circular, el valor de X ha de ser 1,4 veces el valor de Y para el modo HIRES y 1,6 veces en el modo MULTI. En los demás casos la curva resultante no es un círculo sino una elipse.

Ensaya a variar estos dos valores, pero recuerda que tienes una resolución máxima de 320 pixels en dirección horizontal y 200 en vertical para el modo HIRES, y la mitad de resolución horizontal en el modo MULTI. La suma del primer valor que figura a continuación del comando ARC, más el valor del radio de la curva en la dirección X, no debe superar el límite horizontal. Lo mismo se puede decir

para la dirección vertical. Pero surge un problema: como habrás visto, el valor de un byte puede ser mayor que 200. En ambos casos, horizontal y vertical, la superación de este límite hace que la curva discurra a lo largo del borde de pantalla hasta que alcance de nuevo un valor permitido. Cuando esto ocurre continúa de nuevo su camino con normalidad.

El último número que figura en la línea del comando es el valor correspondiente al tipo de dibujo a realizar. Si en este valor pones un 0, borrará un punto, lo cual obviamente no es muy útil. Si pones un 1, se dibujará un punto y con un 2 se producirá una inversión, es decir, desactivará o pondrá en «off» un punto que estaba activado o en «on» y viceversa. Sustituye por un 2 el valor que figura en el programa anterior y verás lo que sucede.

El siguiente comando, ANGL, se usa para dibujar el radio de un círculo y puede resultar muy útil de muchas formas para dibujar cosas tales como los radios de una rueda o las palas de un ventilador. El comando se usa de la manera siguiente (tienes que hacer antes un NEW en tu ordenador):

```
1Ø HIRESØ,1:FORN=
   ØTO36ØSTEP4
2Ø ANGL 16Ø,8Ø,N,84,6Ø,1
3Ø NEXT:PAUSE1Ø
```

También aquí se incluye una varia-

ble como parte de un bucle FOR ... NEXT; con ello se consigue que el programa haga muchas cosas pero con poco esfuerzo de programación. El primer par de valores que sigue a la palabra ANGL son las coordenadas X e Y del centro del círculo cuyo radio se quiere pintar; naturalmente nos estamos refiriendo al punto de partida de dicho radio. El valor N es el ángulo de la perpendicular al radio. Así con un valor de 45 resultaría una línea desde el punto inicial hasta la posición de las tres en punto. Como puedes ver, vale cualquier valor entre 0 y 360. Ensaya a cambiar los valores del incremento STEP de 1 a 10, y a poner un valor fijo de N en la línea 20.

Los tres valores restantes desempeñan una función exactamente igual que en el comando ARC. Te será útil observar lo que sucede con el valor de «borrado» 0. Añádele estas líneas a tu programa actual:

```
3Ø NEXT
35 CIRCLE 16Ø,8Ø,84,6Ø,1
4Ø FOR N=36Ø TO Ø STEP -4
45 ANGL 16Ø,8Ø,N,84,6Ø,Ø
5Ø NEXT:PAUSE 1Ø
```

Lo que hace es repetir el dibujo, añadir un círculo y, en seguida, la línea 45 borra el trabajo hecho por la línea 20.

BLOQUES DE COLOR

El comando BLOCK te permite la creación de rectángulos coloreados. También podrías hacer esto utilizando REC y PAINT, pero con frecuencia es más conveniente usar BLOCK, especialmente cuando se requieren varios rectángulos. El comando toma la siguiente forma (primero tienes que hacer NEW):

```
1Ø HIRES Ø,1:MULTI 2,5,6
2Ø BLOCK 1Ø,1Ø,3Ø,3Ø,1
5Ø PAUSE 1Ø
```

Al ejecutar este programa, te aparecerá un rectángulo de un solo color. El primer par de valores sirve para definir el vértice superior izquierdo del rectángulo, mientras que el siguiente par define el vértice inferior derecho. La última cifra selecciona el tipo de dibujo elegido, lo que en modo MULTI (seleccionado en la línea anterior) corresponde a la primera opción de color, que es el número 2 o rojo. Prueba a hacer algún cambio en el valor del tipo de dibujo, por ejemplo cambiándolo por un 3 y vuelve a ejecutar el programa. O, mejor todavía, añade estas nuevas líneas:

```
3Ø BLOCK 2Ø,2Ø,5Ø,5Ø,2
4Ø BLOCK 3Ø,3Ø,7Ø,7Ø,3
```

Al ejecutar ahora el programa te resultarán tres bloques de color. Para tener unos resultados más interesantes, introduce los siguientes cambios, que harán que aparezca una secuencia aleatoria de quince bloques de colores rojo, verde y azul:

```
2Ø N=5:M=1Ø:C=1
3Ø BLOCK N,N,M,M,C
4Ø N=N+5:M=M+1Ø:C=
```

```
    INT(RND(1)*3+1)
5Ø  IF M<16Ø THEN GOTO 3Ø
6Ø  PAUSE 1Ø
```

MAS DIBUJOS

Los comandos DRAW y ROT se utilizan juntos para dibujar y representar una forma determinada. Se usa una secuencia de instrucciones numeradas en la forma que veremos a continuación (teclea primero NEW):

```
1ØØ  DRAW A$, 15Ø,15Ø,1
```

El componente A$ de esta instrucción define en realidad la forma del dibujo, los dos valores siguientes son las coordenadas X e Y del punto de partida y la última cifra corresponde al tipo de dibujo, igual que antes. La cadena A$ puede contener números del 1 al 9 cuyos valores tienen el siguiente significado:

0 desplazamiento de un pixel hacia la derecha
1 desplazamiento de un pixel hacia arriba
2 desplazamiento de un pixel hacia abajo
3 desplazamiento de un pixel hacia la izquierda
4 no válido
5 desplazamiento a la derecha y dibujo de un pixel
6 desplazamiento hacia arriba y dibujo de un pixel
7 desplazamiento hacia abajo y dibujo de un pixel
8 desplazamiento hacia la izquierda y dibujo de un pixel
9 fin del dibujo

En la cadena A$ puedes utilizar cualquier combinación de números hasta un máximo de 74 por línea; esta limitación tiene por objeto dejar sitio para los otros parámetros que intervienen en el comando DRAW. La construcción de dibujos complicados requiere el empleo de más instrucciones de dibujo de las que caben en una línea, pero puedes encadenar sin problemas varias cadenas de caracteres hasta tener una longitud máxima de línea de 255 caracteres. Un ejemplo de esto es el siguiente:

```
A$ = «UNA LINEA DE 74
CARACTERES»
A$ = A$+ «AUN MAS
CARACTERES»
```

Como habrás visto, sirve para alargar considerablemente A$. Veamos ahora un ejemplo de A$. Teclea lo siguiente:

```
1Ø  A$=«55555555555555555
    565756575555555550163
    788888888888888888888
    888888»
2Ø  A$=A$+«1ØØØØØØØØØØØØØ
    555Ø6881555Ø688655188
    883655551888883655555
    ØØ63»
3Ø  A$=A$+«88888888888888
    8811Ø5555555555755555
    Ø6388886555550688888
    8865»
4Ø  A$=A$+«555555ØØØ68838
    1Ø55551889»
```

Este programa define un esquiador que va descendiendo por una ladera. Pero se requiere algo de programación adicional para representar A$ y aquí es donde entra en juego el comando ROT.

ROT te permite especificar la orientación y el tamaño de la forma definida por el comando DRAW. Añádele ahora la siguiente línea a tu programa:

```
2ØØ  ROT Ø,1
```

El primer parámetro que sigue al comando ROT especifica la amplitud de giro del dibujo. Los siguientes valores muestran las orientaciones posibles:

0 giro de 0 grados
1 giro de 45 grados
2 giro de 90 grados
3 giro de 135 grados
4 giro de 180 grados
5 giro de 225 grados
6 giro de 270 grados
7 giro de 315 grados

Cuando el programa esté listo para ser ejecutado, podrás examinar todas estas alternativas en la línea 200.

El siguiente valor que figura en el comando ROT especifica el tamaño del dibujo. Con el valor 1, el dibujo se hace a tamaño normal. Al ir aumentando este valor, el dibujo se va ampliando, pero debes evitar «superar» la zona disponible de pantalla. Para que el programa se ejecute adecuadamente, introduce estas líneas:

```
 5Ø  HIRES Ø,1
25Ø  PAUSE 2Ø
```

Ejecuta varias veces el programa, introduciendo cambios allí donde sea posible. Siempre puedes reejecutar el programa cuando te falle algún efecto que quieras conseguir.

Incorpora ahora en tu programa las siguientes líneas, para ver la forma de utilizar un grupo de comandos del BASIC de Simons:

```
 5Ø  HIRES Ø,1:MULTI2,5,
     3:COLOUR6,1
 6Ø  LINE Ø,2Ø,32Ø,6Ø,1
 65  LINE32,1,8Ø,7,1
 7Ø  LINEØ,1Ø,32,1,1
 8Ø  LINE25,25,1ØØ,1,1
 85  LINE1ØØ,1,25Ø,5Ø,1
 9Ø  CIRCLE 15Ø,2Ø,5,6,1
 95  PAINT5,5,3
115  FORN=1TO5:READX1,X2,
     X3,X4,X5
12Ø  FORZ=1TOX1STEP4
125  LINEX4,X5+Z,X3,(X5-
     2Ø)+Z,2+RND(1)*2
13Ø  LINEX2,X5+Z,X3,(X5-
     2Ø)+Z,2+RND(1)*2:NEXT
14Ø  BLOCKX3,X5+X1-2Ø,X3,
     X5+X1,1:NEXTN:ROTØ,1
2ØØ  LOW COL 1,4,
     2:FORZ=25TO6ØSTEP3:
     DRAWA$,Z-3,163,1
2Ø5  DRAWA$,Z,163,2:NEXTZ
21Ø  DRAWA$,25,169,3
25Ø  PAUSE 2Ø
1ØØØ  DATA5Ø,33,4Ø,47,9Ø
1ØØ2  DATA2Ø,5,1Ø,15,3Ø
1ØØ4  DATA25,15,2Ø,25,6Ø
1ØØ6  DATA12Ø,1,1Ø,19,9Ø
1ØØ8  DATA2Ø,85,9Ø,95,3Ø
```

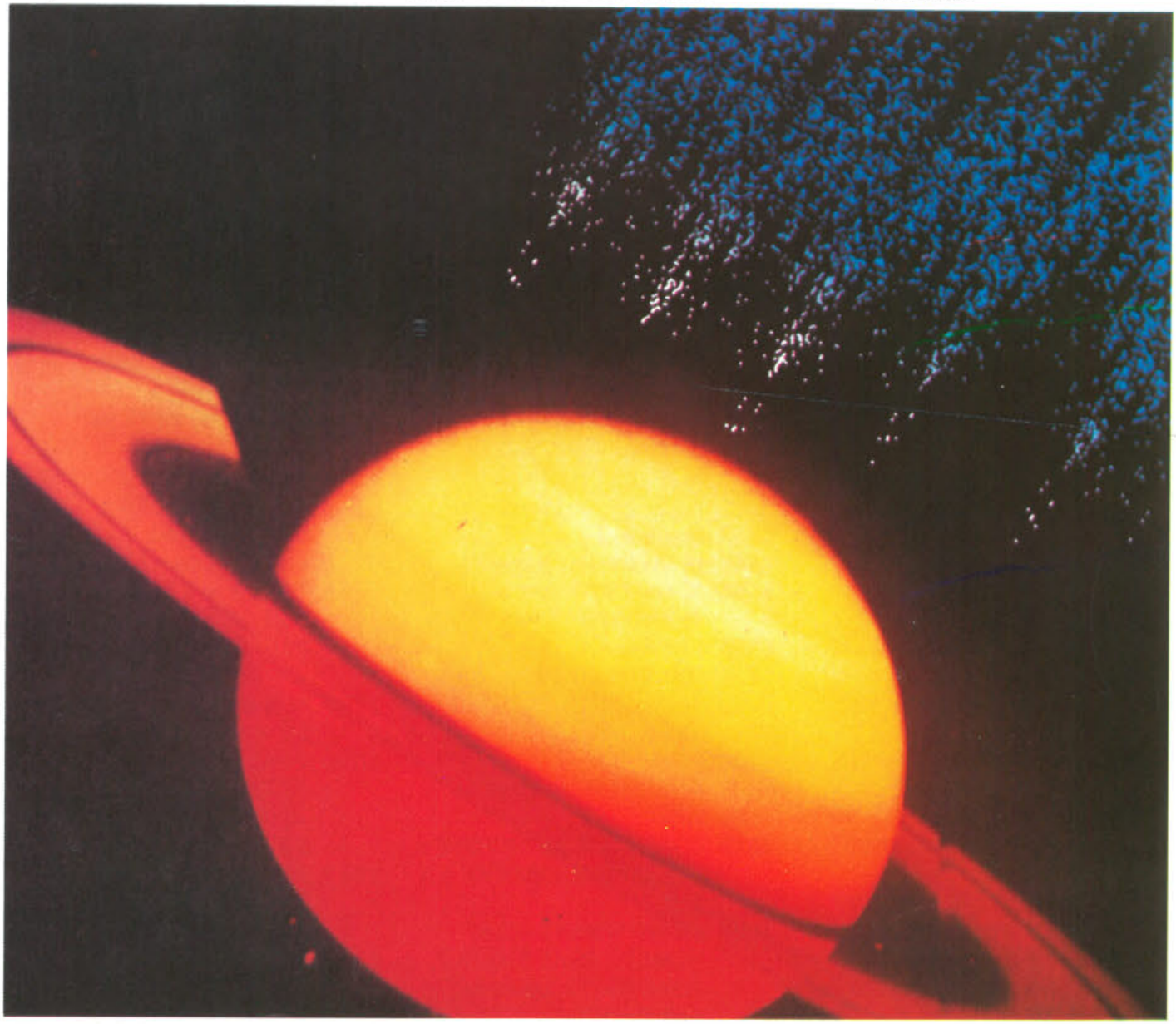
PROYECTO GIOTTO (CONT.)

Tal y como os prometimos en el número anterior, continuamos publicando el interesante listado del "Proyecto Giotto". Como habréis ido comprobando el programa contiene una gran cantidad de informaciones y de opciones, lo que lo hace inusitadamente largo. Para no cansar a nuestros lectores hemos decidido fraccionarlo en varias partes que iremos publicando en números sucesivos. Esperamos que aprendáis mucho sobre el Sistema Solar.

```
1Ø2Ø PRINT "[SHIFT+b]
     [CTRL+6] [ 7 espacios]a [
     17 espacios]b[ 12
     espacios][CTRL+8][
     2*SHIFT+b]"
1Ø3Ø PRINT "[CRSR izqda.]
     [SHIFT+b]";: PRINT
     "[SHIFT+b][ 3 espacios]
     [CTRL+7]1[CTRL+2] ";
     t$(1);"[ 2 espacios][CTRL
     +7]1[CTRL+2] ";u$(1);"
     [ 2 espacios][CTRL+8]
     [ 2*SHIFT+b]"
1Ø4Ø PRINT "[CRSR izqda.]
     [SHIFT+b]"; :PRINT
     "[SHIFT+b][ 3 espacios]
     [CTRL+7]2[CTRL+2] ";
     t$(2);"[ 5 espacios][CTRL
     +7]2[CTRL+2] ";u$(2);"[
     9 espacios][CTRL+8][
     2*SHIFT+b]"
1Ø5Ø PRINT "[CRSR izqda.]
     [SHIFT+b]"; :PRINT
     "[SHIFT+b][ 3 espacios]
     [CTRL+7]3[CTRL+2] ";
     t$(3);"[ 4 espacios][CTRL
     +7]3[CTRL+2] ";u$(3);"[
     14 espacios][CTRL+8] [
     2*SHIFT+b]"
1Ø6Ø PRINT "[CRSR izqda.]
     [SHIFT+b]";:PRINT
     "[SHIFT+b][ 3 espacios]
     [CTRL+7]4[CTRL+2] ";
     t$(4);" [CTRL+7][ 4
     espacios]4[CTRL+2] ";
     u$(4);"[ 17 espacios]
     [CTRL+8] [ 2*SHIFT+ b]"
1Ø7Ø PRINT "[CRSR izqda.]
     [SHIFT+b]";:PRINT
     "[SHIFT+b][ 3 espacios]
     [CTRL+7]5[CTRL+2] ";
     t$(5);"[ 3 espacios][CTRL
     + 7]5[CTRL+2] " ;u$(5);
     "[ 13 espacios][CTRL+8] [
     2* SHIFT+b]"
1Ø8Ø PRINT "[CRSR izqda.]
     [SHIFT+b]";:PRINT
     "[SHIFT+b][ 3 espacios]
     [CTRL+7]6[CTRL+2] ";
     t$(6);"[ 3 espacios][CTRL
     +7]6[CTRL+2] ";u$(6);"[
     13 espacios][CTRL+8] [
     2*SHIFT+b]"
1Ø9Ø PRINT "[CRSR izqda.]
     [SHIFT+b]";:PRINT
     "[SHIFT+b] [3 espacios]
     [CTRL+7]7[CTRL+2] ";
     t$(7);"[ 5 espacios] [CTRL
     +7]7[CTRL+2] ";u$(7);"[
     2 espacios][CTRL+8] [
     2*SHIFT+b]"
11ØØ PRINT "[CRSR izqda.]
     [SHIFT+b]";: PRINT
     "[SHIFT+b][ 3 espacios]
     [CTRL+7]8[CTRL+2] ";
     t$(8);"[ 3 espacios] [CTRL
     + 7]7[CTRL+2] ";u$(8);
     "[CTRL+8][ 2*SHIFT+b]"
111Ø PRINT "[CRSR
     izqda.][SHIFT+b]";:PRINT
     "[SHIFT+b][ 3
     espacios][CTRL+7]9[CTRL+2]
     ";t$(9);"[ 4
     espacios][CTRL+7]9[CTRL+2]
     ":u$(9);"[CTRL+8][ 15
     espacios][SHIFT+b]";
112Ø PRINT "[COMM+g]";
113Ø FOR e=1 TO 38:PRINT
     "[SHIFT+*]";:NEXT
114Ø PRINT "[COMM+w]";
115Ø PRINT "[SHIFT+b][ 13
     espacios][CTRL+6]c
     [CTRL+2] finalizar [CTRL
     +8][ 13 espacios] [SHIFT
     +b]";
116Ø PRINT "[COMM+z]";: FOR
     e=1 TO 38:PRINT
     "[SHIFT+*]";:NEXT :
     PRINT "[COMM+x]";:
     POKE 198,Ø
117Ø INPUT "[CTRL+2] [CRSR
     dcha.][CTRL+9]opcion
     [CTRL+Ø][CTRL+6]";a$:IF
     a$<>"a" AND a$<>"b"
     AND a$<>"c" THEN99Ø
118Ø IF a$<>"a" AND
     a$<>"b" THEN145Ø
119Ø PRINT "[CTRL+2][CRSR
     arriba][ 19*CRSR dcha.]";
     :INPUT "[CTRL+
     9]numero[CTRL+Ø]
     [CTRL+7]";b$
12ØØ IF VAL(b$)>9 THEN 99Ø
121Ø cv$=u$(VAL(b$)):zx=VAL
     (b$)
122Ø IF a$="a" THENnr=1:ON
     zxGOSUB 221Ø,236Ø,
     25ØØ, 297Ø,311Ø,326Ø,
     342Ø, 357Ø,371Ø
123Ø IF a$="b" THENGOSUB
     125Ø:GOTO 99Ø
124Ø GOTO 99Ø
125Ø POKE 5328Ø,Ø:POKE
     53281,Ø:PRINT "[SHIFT
     +CLR/HOME]"
126Ø IF zx<>5 AND zx<>7
     AND zx<>8 THEN132Ø
127Ø IF zx=8 THENcv$="[CTRL
     + 6]revolucion[CTRL+2]
     (en días medios)"
128Ø PRINT TAB(2Ø - (LEN(cv$)
     / 2));cv$
```

```
129Ø IF zx=8 THENPRINT
     "[CTRL+6]":PRINT "[ 19
     espacios]sidereal[ 2
     espacios] sinodica": GOTO
     133Ø
13ØØ IF zx=5 THENPRINT
     "[CTRL+6]":PRINT "[ 16
     espacios]tierra=1[ 3
     espacios]agua=1": GOTO
     133Ø
131Ø IF zx=7 THENPRINT
     "[CTRL+6]":PRINT "[ 19
     espacios]d[ 2 espacios]h[ 2
     espacios]mn[ 2
     espacios]s":GOTO 133Ø
132Ø PRINT "[CTRL+6]":
     PRINT TAB(2Ø - (LEN(cv$)
     / 2));cv$
133Ø PRINT "[CTRL+8][CRSR
     abajo][ 3 espacios]";:
     PRINT t$(1);"...:";"[CTRL
     +2][ 3 espacios]";me$(zx)
134Ø PRINT "[CTRL+8][CRSR
     abajo][ 3 espacios]";:
     PRINT t$(2);".....:";
     "[CTRL+2][ 3 espacios]";
     ve$(zx)
135Ø PRINT "[CTRL+8][CRSR
     abajo][ 3 espacios][CTRL
     +9]";:PRINT t$(3);".....:";
     "[CTRL+2][ 3 espacios]";
     ti$(zx)
136Ø PRINT "[CTRL+8][CRSR
     abajo][3 espacios]";:
     PRINT t$(4);".....:";
     "[CTRL+2][ 3 espacios]";
     ma$(zx)
137Ø PRINT "[CTRL+8][CRSR
     abajo][ 3 espacios]";:
     PRINT t$(5);"....:";
     "[CTRL+2][ 3 espacios]";
     ju$(zx)
138Ø PRINT "[CTRL+8][CRSR
     abajo][ 3 espacios]";:
     PRINT t$(6);"....:";
     "[CTRL+2][ 3 espacios]";
     sa$(zx)
139Ø PRINT "[CTRL+8][CRSR
     abajo][ 3 espacios]";:
     PRINT t$(7);".....:";
     "[CTRL+2][ 3 espacios]";
     ur$(zx)
14ØØ PRINT "[CTRL+8][CRSR
     abajo][ 3 espacios]";:
     PRINT t$(8);"....:";"[CTRL
     +2][ 3 espacios]";ne$(zx)
141Ø PRINT "[CTRL+8][CRSR
     abajo][ 3 espacios]";:
     PRINT t$(9);".....:";
     "[CTRL+2][ 3 espacios]";
     pl$(zx)
142Ø PRINT "[CRSR abajo] [
     12*CRSR dcha.] [CTRL+
     4] [CTRL+9]Pulsa una
     tecla"
143Ø GET a $:IF a$="" THEN
     143Ø
144Ø RETURN
145Ø PRINT "[SHIFT+CLR/
     HOME][CTRL+2]": POKE
     5328Ø,1:POKE 53281,6
146Ø GOTO 184Ø
147Ø POKE 53280,Ø:POKE
     53281,Ø:FOR p=s TO s +
     24:POKE P, Ø:NEXT
148Ø POKE s + 24,15: POKE s
     + 5,15:POKE s + 4,pa:
     POKE s + 6,15
149Ø FOR k=1 TO 7
15ØØ IF K>1 THENj=5:n=4:
     GOTO 152Ø
151Ø mk=1:GOSUB 194Ø: j=35
152Ø FOR a=1 TO j:g=g -
     n:g1=g1 - n:g2=g2 -
     n:g3=g3 - n:g4=g4 -
     n:g5=g5 - n:g6=g6 - n
153Ø re=re + 1
154Ø IF re=35 THEN re=7
155Ø IF re>7 AND re< 9
     THENGOTO 157Ø
156Ø POKE s + 1, re
157Ø ON kGOSUB 163Ø,164Ø,
     165Ø,166Ø,167Ø,168Ø,
     169Ø, 169Ø
158Ø NEXT
159Ø mk=2:GOSUB 195Ø
16ØØ NEXT
161Ø POKE s + 4,pa - 1:POKE s
     + 5,Ø:POKE s,Ø:POKE s +
     1,Ø
162Ø PRINT "[SHIFT+CLR/
     HOME]": POKE 56576,7:
     POKE 53272,2Ø:POKE
     648,4:END
163Ø POKE v,x:POKE v + 1,g
164Ø POKE v + 2,x:POKE v + 3,
     g1
165Ø POKE v + 4,x:POKE v + 5,
     g2
166Ø POKE v + 6,x:POKE v + 7,
     g3
167Ø POKE v + 8,x:POKE v + 9,
     g4
168Ø POKE v + 1Ø,x:POKE v +
     11,g5
169Ø POKE v + 12,x:POKE v +
     13,g6
17ØØ POKE v + 14,137:POKE v
     + 15,g6
171Ø RETURN
172Ø POKE 56334,
     PEEK(56334) AND 254:
     POKE 1; PEEK(1) AND
     251: SYS 49152,53248,
     55296,36864
173Ø POKE 1, PEEK(1) OR 4:
     POKE 56334,
     PEEK(56334) OR 1:POKE
     53272,(PEEK(53272) AND
     24Ø) OR 2
174Ø FOR Pa=1 TO 34:READ
     car:FOR byte=Ø TO 7:
     READ a: POKE 34816 + (8
     * car) + byte,a
175Ø NEXT byte: NEXT pa
176Ø g=77:g1=98:g2=119:g3
     =14Ø:g4=161:g5=182:g
     6=2Ø3:1=1ØØØ:n=2:v=
     53248:x=161
177Ø s=54272:pa=129:re=Ø
178Ø POKE v + 21,255: POKE v
     +21,255:POKE v +28,
     255: POKE v + 37,Ø:POKE
     v + 38,12
179Ø FOR c=Ø TO 7:POKE v +
     39 + c,1:NEXT
18ØØ POKE 33784,16:POKE
     33785,17:POKE 33786,
     18:POKE 33787,19: POKE
     33788,2Ø
181Ø POKE 33789,21:POKE
     3379Ø,22:POKE 33791,
     23
182Ø FOR z=Ø TO 7:FOR p=pr
     TO pr + 62: READ a:POKE
     p,a:NEXT:pr=pr + 64:
```

```
    NEXT z
183Ø GOTO 75Ø
184Ø POKE v,x:POKE v +1,g:
    POKE v + 2,x:POKE v + 3,
    g1:POKE v + 4,x:POKE v
    + 5,g2
185Ø POKE v + 6,x:POKE v + 7,
    g3:POKE v + 8,x:POKE v
    + 9,g4:POKE v + 1Ø,x:
    POKE v + 11,g5
186Ø POKE v + 12,x:POKE v +
    13,g6:POKE v + 14,137:
    POKE v + 15,g6
187Ø PRINT "[CTRL+2][CLR/
    HOME][ 3*CRSR abajo]"
188Ø FOR c=1 TO 2Ø:IF c=1
    THENPRINT TAB(2Ø);
    vc$(c):GOTO 193Ø
189Ø IF c=2 THENPRINT
    TAB(19);vc$(c):GOTO
    193Ø
19ØØ IF c=18 THENPRINT
    TAB(16); vc$(c): GOTO
    193Ø
191Ø IF c=19 OR c=2Ø
    THENPRINT TAB(14);
    vc$(c):GOTO 193Ø
192Ø PRINT TAB(2Ø);vc$(c)
193Ø NEXT
194Ø IF mk=1 THENPRINT
    "[CTRL+8]";
195Ø IF mk=2 THENPRINT
    "[CTRL+4]";
196Ø IF mk=Ø THENPRINT
    "[CTRL+2]";
197Ø PRINT "[CLR/HOME][
    19*CRSR abajo][SHIFT+
    d][ 2*SHIFT+f][SHIFT
    +d][SHIFT+e][
    6*COMM+g][ 3*SHIFT +
    e][ 5*COMM+g][
    2*SHIFT+e][ 4*CRSR
    dcha.][ 2*COMM+g]
    [SHIFT+e][ 2*SHIFT+ d][
    4*SHIFT+f][ 2*SHIFT+d][
    3*SHIFT+ f][SHIFT+*]"
198Ø IF mk<>Ø THEN
    RETURN
```



EL ZOCO

Cambio juegos para C-64, no importa que tengas pocos, escríbeme y te prometo contestar; tengo últimas novedades.
Oscar Fernández Orallo, Plaza la Fortaleza n.º 11 4.º C 24400 Ponferrada. León.

Vendo ordenador Commodore 128. Incluyo en el precio cassette C2N, Joystick Spectravideo, 140 programas en cinta, manual del usuario, 35 revistas especializadas, juegos en cinta, tres libros DATA BECKER (el ensamblador, 64 consejos y trucos) Simon's Basic, disco con el sistema operativo CP/M PLUS.
Todo por 70.000 pts. NEGOCIABLES. Javier Antón Morán. C./ El Milagro n.º 26. 39300 Torrelavega. Cantabria. Tl: (942) 88 12 54.

Intercambio programas y experiencias con usuarios del C-16. Juan Antonio Domínguez Zafra C./ Blas Infante n.º 12 izq 21006 Huelva. Tl: (955) 23 30 87.

Vendo ordenador SHARP MZ-721 de 64k, con 130 juegos y aplicaciones por 50.000 pts., cassette incluido. Tl: (93) 340 50 87 Víctor.

Vendo/cambio programas para C.64 a todo nivel, poseo más de 200 y novedades. Ofertas económicas en juegos buenos (menos de 600 ptas.). Noé Santalluria Ramos C./ Academia n.º 46 1.ª A 25002 Lérida. Tl: (973) 26 42 48.

Vendo interesante programa educativo, interactivo con el usuario. Nivel COU-3.º BUP. Tema: Genética. Magníficos gráficos y carga con Turbo. Precios especiales para colegios.
Germán Rodrigo García C./ Chiva n.º 11. 46380-Cheste (Valencia). Tl: (96) 251 05 80. Llamar preferiblemente por las tardes.

Vendo monitor en color Philips por sólo 30.000 pts. Llamar al tl: (93) 203 21 64 Cristina.

Cambio programas de juegos (preferentemente de calidad) en cinta para Commodore 64.
Jaime Bertrán Martínez
C./ La Baronía n.º 42. 08208 Sabadell. Tl. 717 36 85.

Intercambio programas en disco, para C-64. Escribir a: Juan Jesús Miranda Ortega, Barriada Sta. Paula blg San Pedro 15 piso B5. 29004 Málaga. Tl: 34 62 93.

C.I.C. (Club Intercommodore), pedir información a C.I.C. calle Algorta n.º 9, Buzón 9. 28019 Madrid. Porque un CLUB sirve para algo más que para intercambiar juegos.

¡Atención!: Zx Spectrum Plus, con todos sus accesorios y manuales, más interface Kempston y Joystick Quick Shot II, todo en perfecto estado y con embalajes originales; revistas Microhobby (colección completa) con cursos de Basic y C/M, juegos originales de calidad y curso de Basic de Investrónica. *Todo* a un precio estupendo. Si te interesa, llama al teléfono (976) 51 07 15, de Zaragoza.

GANADORES DE LOS MEJORES DE INPUT COMMODORE

En el sorteo correspondiente al número 14 entre quienes escribisteis mandando vuestros votos a LOS MEJORES DE INPUT han resultado ganadores:

NOMBRE	LOCALIDAD	JUEGO ELEGIDO
María Soledad Cejas Gómez	Málaga	Parallax
Pedro Castaño Pérez	H. de Llobregat (Barna.)	Uridium
Javier Ansoleaga Gil	Logroño (La Rioja)	Winter Games
Antonio Portero Angulo	Zaragoza	Spindizzy
Enrique Sarti González	Seseña (Toledo)	Knight Games
Enrique Pérez Intxaurrandieta	Rentería (Guipúzcoa)	Green Beret
Roberto Mateo Gañan	Burgos	Knigat Rider
Jesús García Martínez	Guadix (Granada)	Green Beret
Daniel Fdez. de la Oliva	Majadahonda (Madrid)	Mission Elevator
Angel Díaz Miguel	Madrid	THANATOS

SOFTACTUALIDAD

TAU CETI O LA AVENTURA ESPACIAL

Tau ceti forma parte de un nuevo concepto de juegos: un tipo de juegos nuevo en el que se combinan de un modo espectacular la acción y la aventura en un marco espacial que emula una conquista galáctica tipo STAR WARS. Se trata de llegar a dominar unos planetas usando y abusando de dinero, influencias y, cómo no, de los tiros. Este es un juego que encantará a los que amaron el **PSI V trading COMPANY de ACCOLATE.** Hay ventanas que se abren y cierran sin cesar haciendo que la acción se desarrolle en varias pantallas superpuestas.

★★★★★★★★★★★★

FREAK FACTORY

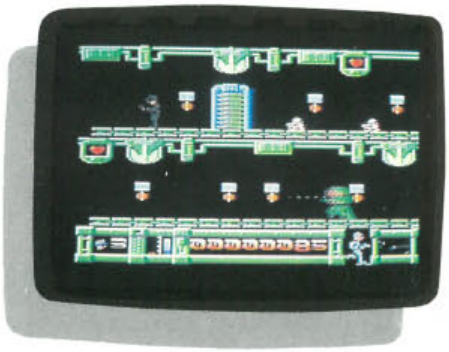
Freak factory será sin duda el programa predilecto de todos aquellos que amaron la casa que creó **SIR ARTHUR PENDRAGON;** los gráficos parecen más cuidados y el sonido está sensiblemente mejorado con respecto a aquella casa.

Quizás el número de personajes que intervienen sea sensiblemente menor que en el caso de los juegos que le precedieron, pero el número de pantallas (267 según nos dicen) hará olvidar este pequeño detalle. A recomendar a los amantes de los juegos de salas.

★★★★★★★★★★★★

MISSION AD

En principio el nombre puede sonar a juego de aventuras. Pero no se trata de un juego endiabladamente difícil, que pretende hacerte perder la paciencia mientras descubres unos objetos en medio de una ciudad. Los gráficos son, cuando menos, brillantes; el sonido es sin duda lo



mejor del juego (excelente). La acción se desarrolla en unos pasillos plagados de enemigos dignos de la cafetería de **George Lucas**. Por medio de ascensores lograrás pasar una serie de zonas y visitar las innumerables salas que tiene esta ciudad.

★★★★★★★★★★★★

THE FOURT PROTOCOL

Tal y como el nombre del juego nos indica, en este caso estamos ante una aventura de espionaje basada en el libro de Frederic Forsyth, **El**

Cuarto Protocolo. El programa se nos va presentando a través de menús, similares a los utilizados por los ordenadores personales de APPLE, lo que establece una diferencia importante con el resto de los juegos de espionaje, la partida se desarrolla sin necesidad de escribir con comandos de textos. En su lugar se emplean los iconos, a través de los cuales elegimos los caminos a emprender y nuestras **artes** para camuflarnos e impermeabilizarnos frente a los perseverantes **servicios de inteligencia**. El número de niveles es de tres, si bien es obligatorio comenzar por el primero y superar las pruebas para poder ir ascendiendo al segundo y, finalmente, al tercero.

★★★★★★★★★★★★★

PROFIPAINTER

La conocida empresa Data Becker ha comercializado un programa para dibujo con numerosas e interesantes novedades. Al igual que el comentado inmediatamente antes, a través de una exposición de menús superpuestos a los que se accede mediante ventanas, ofrece diferentes opciones de relleno. Utilizando un lápiz desplazable a través de la pantalla escogemos los tipos y símbolos con los que queremos dibujar. El programa está capacitado para trabajar con imágenes digitalizadas que se pueden adquirir en el mercado. Como ampliación, dispone de cuatro juegos de caracteres distintos a los de la ROM original. Asimismo, se asemeja mucho a las utilidades y funciones de los grandes programas de dibujo industrial.

LA REDACCION CAMBIA DE DIRECCION

ESTAMOS

Aribau
n.º 185
planta, 1
08021
Barcelona

INFILTRATOR

DATOS GENERALES

TITULO Infiltrator

FABRICANTE US Gold

ORDENADOR C-64

TEMA DEL PROGRAMA Aventuras

CALIFICACION (Sobre 10 ptos.)

ORIGINALIDAD	8
INTERES	8
GRAFICOS	9
COLOR	9
SONIDO	6
TOTAL	40

El héroe de **Infiltrator**, el llamado **JIMBO BABY**, parece un hombre poco común: «supersoldado, piloto de helicóptero, cirujano, ingeniero, experto en balística, político, orador, estrella de rock, explorador, etc...» Uno se identifica en seguida con el personaje eso sí!!!

Se supone que hay un loco que anda suelto por ahí y eres el único en todo el planeta que puede resolver una situación que está rompiendo el equilibrio vital del sistema.

El estado mayor pone a tu disposición un helicóptero de la marca GIZMO DHX-1.

El equipo de este helicóptero es sencillamente fantástico: disparador antimisil, sistema de control y vigilancia, vuelo silencioso, cañones de 80 mm, bengalas de magnesio, incluso tu helicóptero tendrá unos asientos de auténtica piel de oso.

El helicóptero sólo se puede armar a partir de los 2000 pies. Dispones de un tridimensional de la zona que ocupas y de los amigos y enemigos.

El ordenador de a bordo te da también un fiel mapa de las zonas dañadas por los enemigos de tu superhelicóptero.

Hasta aquí el juego no deja de ser un excelente simulador de vuelo de helicóptero algo difícil de manejar (lo mejor es que te consigas un copiloto). Y aquí llega la sorpresa: de golpe aterrizas en una especie de llano y el pilotaje deja lugar a un juego de auténtica aventura, acción, espionaje industrial. En efecto, te ves equipado con una serie de objetos, como aparato de fotos, falsos papeles, pulverizador, carga de dinamita, detector de minas, etc. De todo este equipo sólo puedes seleccionar una cosa a la vez; quizás lo mejor sea tomar el detector de minas antes de acercarse a la base que se distingue no muy lejos.

Después de los momentos de pánico que habrás sin duda vivido en el helicóptero ahora te llegan momentos de desesperación en los múltiples decorados de esta epopeya.

Mientras penetras en la base verás que el primer guardia te para y te busca las cosquillas. Sin duda la mejor solución que deberás emplear en muchas ocasiones, sea la de enseñar unos falsos papeles. A veces, el guardia no es tan torpe como parece y deberás entonces hacerle probar ese fabuloso pulverizador que tienes en el bolsillo, sin embargo, si tratas de huir sin haber enseñado tus papeles, o sin haberle hecho dormir la alarma será dada.

La diversidad de pantallas, la soberbia animación, los efectos de sonido, etc. convierten a Infiltrator en un juego recomendable tanto a amantes de acción como a amantes de estrategia.

★★★★★★★★★★★★★★★★★★★★★★★★★★

THE ASSAULT MACHINE

A bordo de la nave interestelar Peacekeeper tienes una importante misión que cumplir.

La historia tiene un origen remoto: hace varios siglos, los habitantes del planeta Targón se dedicaban a la piratería y el saqueo galáctico; a consecuencia de ello, su planeta fue convertido en una prisión y, con sus habitantes, aislado del resto de la Galaxia. Ahora, los **Infiltrados** liberan Targón y se han conjurado para destruir el resto de la Galaxia. Para ello están construyendo un terrible sistema guerrero: **The Assault Machine**, formado por diez complejos militares diseminados por otras tantas islas de las muchas que existen en el planeta.

Y, aquí, apareces tú de nuevo, al mando de tu flamante nave, enviado a explorar Targón para descubrir el mortífero sistema y destruirlo. En primer lugar debes iniciar la exploración. Una vez hayas localizado una isla no explorada enviarás allí a uno de los cuatro **droids** exploradores de que dispones. Para el desembarco, trasladarás el **droid** en una nave atmosférica. Tienes bajo tu mando tres de estas naves, susceptibles de transformarse en precisos bombarderos si tus averiguaciones obtienen un resultado positivo.

No pienses, sin embargo, que podrás dedicarte sin perturbaciones a tus actividades topográficas. Tus **droids** son amenazados y hostigados constantemente por los malévolos habitantes de Targón, que detentan una sofisticada panoplia de armas: submarinos, carros y bombarderos. Tanto en este caso como en el de que hayas localizado alguna parte del nuevo sistema de armas, te verás abocado a una electrizante refriega entre tu nave y las de Targón. Afortunadamente, la información de que dispones es muy completa. Tu tablero de control te indica cuáles son tus reservas de municiones, el estado de tus exploradores, las averías y la velocidad a la que avanzas. Además un radar descubre las posiciones enemigas y el tiempo de que dispones. Este último punto es de vital importancia, pues tu lucha es contra el reloj, cuanto más tiempo ganas, lo que se consigue destruyendo complejos, más difícil lo tienen los **Infiltrados** para acabar su macabra obra.

Este juego te gustará sin duda, pues aúna variedad de escenarios, con unos gráficos de gran calidad y una gran rapidez y precisión de movimientos.

DATOS GENERALES

TITULO The Assault Machine

FABRICANTE Nexus

ORDENADOR C-64/128

TEMA DEL PROGRAMA
Aventura galáctica

CALIFICACION (Sobre 10 ptos.)

ORIGINALIDAD	9
INTERES	8
GRAFICOS	9
COLOR	8
SONIDO	8
TOTAL	42

★★★★★★★★★★★★★★★★★★★★★★★★★★

TWO ON TWO (BALONCESTO)

TWO on TWO es como su nombre indica una simulación de un campeonato de baloncesto de la liga **GBA** de dos en dos.

A nivel de juego la gama de posibilidades que te ofrece es inmejorable: solo contra el ordenador, dos jugadores simultáneos contra el ordenador, un jugador contra otro siendo cada uno asesorado por el ordenador, sesiones de entrenamiento, etc..., hasta alcanzar el definitivo partido de Campeonato de la liga.

Cuando estés descansado con el joystick en la mano antes del partido podrás definir una serie de parámetros de vital importancia para tu juego: así, deberás combinar 6 parámetros que incluyen desde el porcentaje de aciertos dentro y fuera del área, jumping fuerte o no (habilidad para tirar dentro en suspensión y recoger rebotes) y, por último, STEALING a la habilidad

que tienes para que nadie te quite la pelota.

Podrás también realizar un entrenamiento previo antes de los partidos para calentar esos músculos y articulaciones que tanto te permitirán marcar luego.

La opción de juego con dos jugadores es realmente buena, ya que el ordenador te apoya y ayuda en las jugadas de un modo realmente profesional.

La variedad de jugadas permitidas es enorme, ya que se pueden realizar múltiples figuras. Los GANCHOS se realizarán desde la línea de la base, al fondo de la cancha. Los MATES desde las tres zonas (a medio paso a la derecha e izquierda del poste así como debajo de la canasta). Los rebotes, como los tiros, en suspensión desde la línea de tres puntos.

Durante el duro y peliagudo **combate** podrás elegir entre una serie de estrategias de ataque y defensa. Así, podrás optar por defender mediante una lucha hombre a hombre o bien por medio de una defensa por zonas, en la que repartirás zonas para que tú y tu compañero cubráis una zona concreta.

Otro de los detalles es sin duda el peculiar árbitro del encuentro, quien sin la más remota molestia **pitará** faltas personales en ataque, en defensa, TRAVELING —cuando un

DATOS GENERALES

TITULO Basketball

FABRICANTE Activision

ORDENADOR C-64/128

TEMA DEL PROGRAMA Deportes

CALIFICACION (Sobre 10 ptos.)

ORIGINALIDAD	10
INTERES	10
GRAFICOS	8
COLOR	9
SONIDO	7
TOTAL	44

jugador salta para tirar y no realiza la acción a tiempo—, asimismo aplica sin parpadear y con una perfección ejemplar la regla de los tres segundos y castiga con la pérdida de la posesión cualquier retraso de más de 24 segundos. Evidentemente caben también las opciones de tiempos con la petición de tiempo muerto e incluso, colmo del perfeccionismo, repetición de jugadas (esta vez el patrocinador no es la SEAT) opción en la que el programa repetirá las jugadas que le han parecido más acertadas.

Al igual que cualquier profesional de esta liga GBA podrás, al final del partido, ver las estadísticas de juego de los componentes del partido. En ellas encontrarás curiosidades como por ejemplo número de canastas de dos y tres puntos, rebotes, bolas robadas, y, cómo no, máximo encestador.

Por si fuera poco, el juego te da a elegir de entre una base de datos de jugadores de gran calibre de la NBA (esta vez auténtica) como son MAGIC LINDON, LARRY BERG, KAREEM UGRIM, etc..., cada uno se convertirá en tu compañero de equipo en los fantásticos torneos que te esperan.

La conclusión evidente es que

estamos ante una soberbia simulación de este deporte rey que ha cobrado muchos adeptos nuevos en los últimos tiempos: sin duda un juego que dará lugar a horas y horas de entretenimiento tanto a los amantes de este deporte como a los que no son tan amantes.

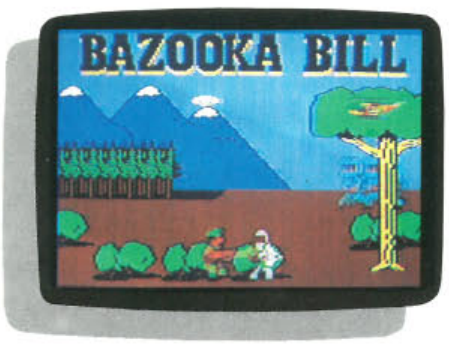
¡ALTAMENTE RECOMENDADO!

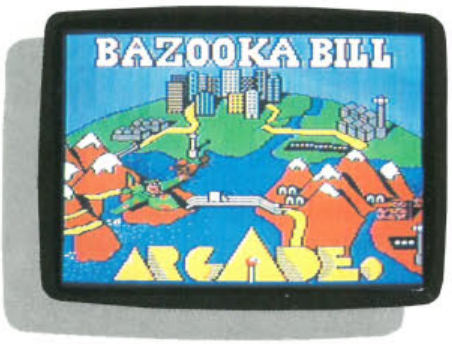
★★★★★★★★★★★★★★★★★★★★★★★★★

BAZOKA BILL

Bazoka Bill es tal y como su nombre indica un mercenario acostumbrado a juegos de guerra. Esta vez ha sido enviado a la base militar de CLARK, donde va a tratar por todos los medios de rescatar a un general hecho preso por unos secuestradores.

Como os podéis imaginar este soldado ha ganado este apodo por su fabulosa habilidad con los bazokas. Aunque también hay que reconocer que tiene puños formidables que más de una vez le han salvado la vida.





En su intento de rescatar al general se ve lanzado en paracaídas en medio de territorio enemigo sin ningún tipo de armas: esta vez la misión va a ser mucho más difícil de lo acostumbrado: deberá hacerse con armas en un camino plagado de peligrosos enemigos.

Como es lógico, el medio más rápido de armarse es cogiendo las armas que los enemigos van dejando: cuchillos, metralletas, lanzallamas, e incluso bazokas, serán las posibles **herramientas** a utilizar, en este juego hay un detalle curioso: la puntuación, mejora de fuerza y vidas

DATOS GENERALES

TITULO Bazoka Bill

FABRICANTE Fitzroy House

ORDENADOR C-64/128

TEMA DEL PROGRAMA
Aventura de guerra

CALIFICACION (Sobre 10 ptos.)

ORIGINALIDAD	6
INTERES	6
GRAFICOS	7
COLOR	8
SONIDO	6
TOTAL	33

adicionales se obtiene por medio de una marcador un tanto macabro: los muertos del campo contrario (cuando alcances más de 100 muertos ganarás una vida).

El argumento del juego está un tanto visto, ya que está en la línea de RAMBOS, WHO DARES WIN. A nivel gráfico está totalmente en la línea del Green Beret aunque de un nivel inferior. El sonido es un poco ronco, pero no tiene porque haber mucho sonido en un juego de esta clase. Juego a recomendar sólo para los incondicionales.

★★★★★★★★★★★★★★★★★★★★★★★★★★

BREAK THRU

BREAK THRU es un programa editado hace ya un cierto tiempo, y que hemos decidido comentar antes de que quede guardado en los cajones para siempre.

Se trata, a pesar de que su nombre no nos lo advierte, de un juego del tipo del MOON PATROL.

Un automóvil que nosotros pilotamos ha de avanzar a través de las líneas enemigas en busca de un avión. Al igual que en el MOON PATROL en este caso te deberás enfrentar a aviones, mientras que los tanques se ocultan en los túneles y utilizan apoyo de lanzallamas. Los campos que atraviesas están sembrados de minas que explosionan al menor contacto.

Tu misión consta de cuatro fases: la avanzada por un campo, el paso del campo, la lucha en la ciudad y por último, el aeropuerto.

En el paso por el campo el mayor de los problemas será elegir la trayectoria más idónea para pasar por encima de un lago. El fin último del juego es llegar sin problemas a un avión evitando un potente lanzallamas enemigo.

El programa pertenece a la casa DATA EST (autora entre otros de KARATE CHAMP, etc.), pero sin embargo, en esta ocasión, el nivel del juego está bastante por debajo de la línea de productos a los que nos tiene acostumbrados esta casa.

DATOS GENERALES

TITULO BREAK THRU

FABRICANTE DATA EST

ORDENADOR C-64

TEMA DEL PROGRAMA
Aventuras de guerra

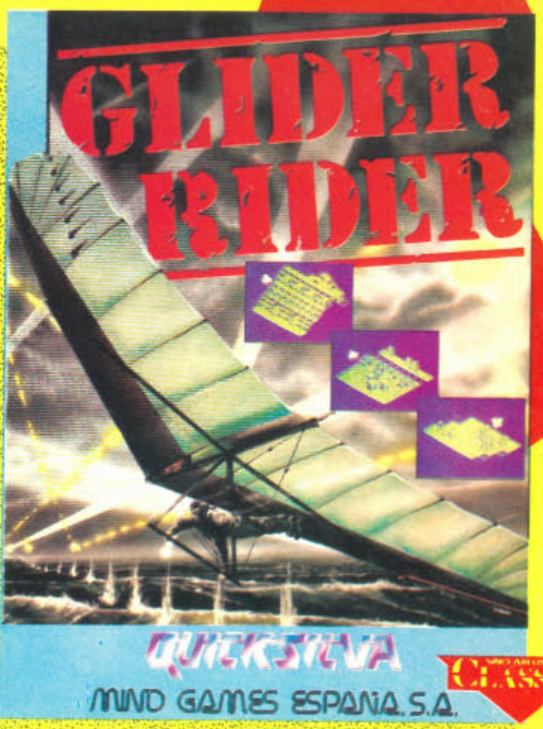
CALIFICACION (Sobre 10 ptos.)

ORIGINALIDAD	5
INTERES	5
GRAFICOS	6
COLOR	7
SONIDO	7
TOTAL	30

¡¡LOS MEGA HITS!!

(C)

Mercenario del espacio... ganas tus armas y escudos defensores de la manera más dura... matando por ellos.
Controlado por iconos, tú cargas tu cinturón de armas, miras la galaxia y eliges desde 100 misiones ofensivas diferentes. Es todo acción con un fuego en 16 direcciones diferentes.



(A) (C) (S)

Esta es una real y verdadera misión diabólica. Como comandante Glenn White, has sido lanzado dentro de la isla artificial de Abraxas Corporación con sólo una moto, un ultraligero y nueve granadas.
Tu misión, destruir los Reactores Nucleares de la isla..., ¡suerte!

Disponibles en:
CASSETTE
y
DISCKETTE

(A) (C) (S)

Parecía un sueño en el año 1986, pero en 2056 la «Strategic Defence Initiative» (Iniciativa para la Defensa Estratégica) ya se había convertido en un hecho.
Ellos llegaron el día 24 de diciembre...
La Fuerza Alienígena se hizo cargo del sistema defensivo y lo convirtió en un arma mortal.
Estás a sólo 160 segundos de la CONDICION ROJA. ¡MUEVETE!

(A) (C)

Intenta ganar el Rally Urbano más duro del mundo compitiendo contra otros 7 pilotos a través de 360 pantallas y durante día, noche y día con sol, lluvia, nieve y tormentas.

AMSTRAD	(A)
Commodore	(C)
SPECTRUM	(S)

VISITE LA DIVISION Online Galerías Preciados

A LA VENTA EN GALERIAS
Marcando estilo.

Y EN TODOS LOS DISTRIBUIDORES DE NUESTROS PRODUCTOS

Editado y distribuido en España por:

MIND GAMES ESPAÑA S.A.

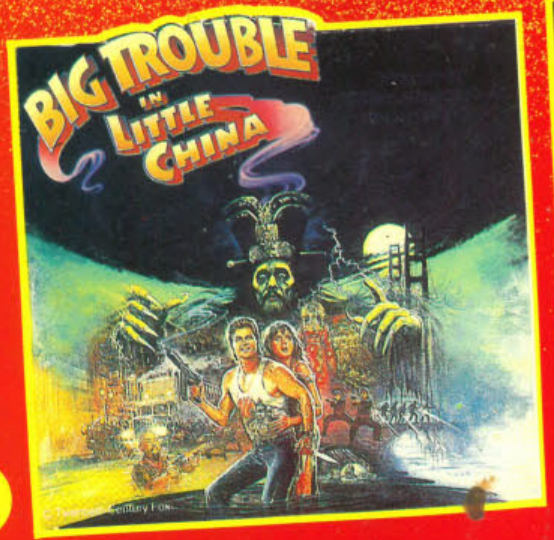
Mariano Cubi, 4 Entlo. Tel. 218 34 00 - 08006 Barcelona

LOS EXITOS DEL CINE
EN TU ORDENADOR

C

CSA.

CSA.



CSA.

CSA.

Disponibles para:

COMMODORE
SPECTRUM
AMSTRAD CASS/DISK

C
S
A

EN TIENDAS ESPECIALIZADAS Y GRANDES ALMACENES, O DIRECTAMENTE POR CORREO O TELEFONO A: **PROEIN, S.A.**

Distribuido en Cataluña por: DISCOVERY INFORMATIC C/. Arco Iris, 75 - BARCELONA - Tels. 256 49 08 / 09

Velázquez, 10 - 28001 Madrid - Tels. (91) 276 22 08/09